

S I N C L A I R

Every month £1.75 September 1989

QL WORLD

DIY TOOLKIT

Fast vector and pixel
graphics

2 to 7 9 10 12 14 17 DISC CHIPS
18 CIRC. MODS. 24 28 32 34 CAPS
39 40 BACK COVER OUTER



MORE DISC DRIVES FROM DENNIS

YOU CANNOT MEAN . . .

Part 2: hints on interpreting
The Editor (Special Edition)

ARTIST OF THE YEAR 1988

At last, the results.

Editor
Helen Armstrong

Chief Sub Editor
Harold Mayes MBE

Production Manager
Nick Fry

Designer
Chris Winch

Advertising Sales
Jason Newman

Magazine Services
Sheila Baker

Advertising Production
Michelle Evans

Publisher
Perry Trevers

Managing Director
Peter Welham

Financial Director
Brendan McGrath

Chief Executive
Richard Hease

Microdrive Exchange
089 283 4783/2952
(2 lines) TIL

Sinclair QL World
Greencoat House
Francis Street
London SW1 1DG
Telephone 01-834 1717
Fax 01-828 0270
Telex 9419564 FOCUS G
ISSN 026806X

Unfortunately, we are no longer able to answer enquiries made by telephone. If you have any comments or difficulties, please write to The Editor, Open Channel, Trouble Shooter, or Psion Solutions. We will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.

Back issues are available from the publisher price £2 U.K., £2.75 Europe. Overseas rates on request. Please telephone 089 283 4783 to check availability.

Published by Focus Magazine Ltd., London. S M Distribution, Streatham, London SW1.

Subscription information from: TIL, PO Box 74, Paddock Wood, Tonbridge, Kent TN12 6DW.

£15.00 U.K., £30 Surface mail Europe and the rest of the world. Add £5 for air mail + £10 overseas.

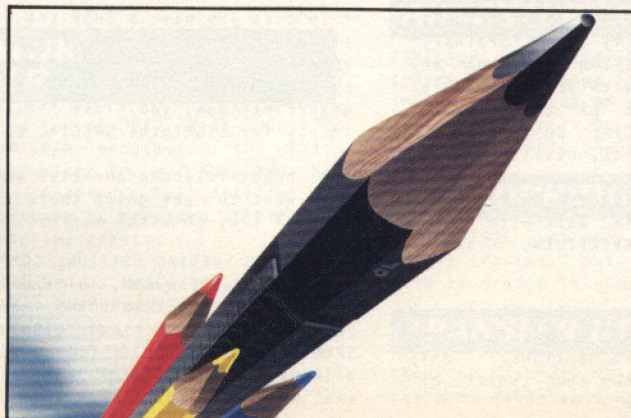
Typesetting by Adtec Typographics, Stanford-le-Hope, Essex.

Tel: (0375) 360967.
Printing by Southernprint Ltd.
©COPYRIGHT SINCLAIR QL
WORLD — 1989.

CONTENTS

SEPTEMBER 1989

- 9 QL SCENE ● QL show in Italy
- 10 OPEN CHANNEL ● SuperBasic erratum
- 11 SUBSCRIPTION INFORMATION
- 12 TROUBLESHOOTER ● How to write for The Progs
- 14 PSION SOLUTIONS ● Quill and printers
- 17 DISC DRIVES ● Another morsel from Dennis
- 18 PAPERWORK TAKES A QUANTUM LEAP ● Happy Friday
- 24 ARCHIVE REVEALED ● Hidden graphics in Archive
- 26 ARTIST OF THE YEAR 1988 ● Head-turning
- 28 YOU CANNOT MEAN... ● The Editor as a wordprocessor
- 32 BASIC IMPROVEMENTS ● Familiarising functions
- 34 DIY TOOLKIT ● Fast vector and pixel graphics
- 39 ONE MAN'S HARDWARE ● A system is born
- 40 SUPERBASIC ● File matching and wild cards
- 44 PROGRAM OF THE MONTH ● Wordblok
- 48 MICRODRIVE EXCHANGE ● Little things are sent to try



NEXT MONTH

ARCHIVE SECRETS

An investigation into some of the undocumented features of Archive.

YOU CANNOT MEAN ...

Another between-the-lines look at a popular QL package.

QL SHOW AT VILLA ALBA

The first Italian QL meeting will take place at the Villa Alba, Gardone Riviera (Brescia), Italy on September 23. The all-day event is expected to include firms exhibiting and selling QL products, with demonstrations and talks by producers and users. All interested users are invited to speak by the organiser, SPEM of Turin, which requests that a script of any talk should be delivered at least a week in

advance, written in English, so that it can be translated into Italian.

Ad hoc offers to speak will be accepted but, says Guido Masoero of SPEM, "the translation could be bug-ridden and slow." Even so, the offer is open. Among the names which SPEM hopes to have attending are Digital Precision, Miracle Systems, Sector Software, PDQL, Jochen Merz Software, Tony Tebby and members of

various European user groups, including Quanta. The villa is a 19th century building by popular Lake Garda, with a restaurant and hoels in the locality and parking for 800 cars.

SPEM clearly wants the convention to be an open and friendly event. More details, including stand bookings, can be obtained from Guido Masoero, SPEM, Via Aosta 86, 10154 Torino, Italy. Tel: 0101 011 857924.

Erratum

There were two small errors in the *Printer Spooler* program in the July issue of *QL World*.

The program should be started by EXEC mdv1_spooler, not EXEC_W mdv1_spooler, and the definition to send CHR\$(10) to the printer should be lf, not lf as printed.

Euroshow

The European Microfair for Sinclair Users, organised by the Club Sinclair BruQsl for QL, Spectrum, Thor, Z-88, ZX-81 and ZX-80 users, will be held at the Eurovolleycenter, Beneluxlaan 22, 1800 Vilvoorde, Brussels, Belgium on October 21.

The exhibition hall is near to exit 6 of the Brussels ring road and close to the international airport. There is a hotel on the site and there will be rooms for lectures as well as exhibitions. A bureau de change will be running on the site for foreign visitors.

More information from Jacques Tasset, secretary, Club Sinclair BruQsl, Aarlenstraat 104, 1040 Brussel, Belgium.

Meet the Fleet

Fleet Tactical Command, an ambitious new real-time networking game for the QL, is due for release soon. Ideally, says the manufacturer Di-Ren, the all-machine code game is played between two computers over network, serial or modem links. It can be run on a single computer for practice.

In real-time and three dimensions, the game is designed to be as realistic as possible. The scenario is set in 1,000 square miles of ocean, with a team of 16 vessels under the guidance of each player. The player is the *Fleet Tactical Commander*, who can move from ship to ship, observing and guiding the action from the bridge of the appropriate vessel.

Completing the realism, each package will contain an instruction manual, two sets each of charts, compass/dividers, protractors, pencils, erasers and rulers for navigation, and two copies of the game on microcassette or disc. Networking or serial-to-serial port leads are optional extras.

Fleet Tactical Command will cost £49.95 for the two-player package. "This is an unusual idea, so it is something of a shot in the dark," says proprietor Robin Barker. He expects the game will become available on other computers, including various Amstrad models, in due course.

Enquiries to Di-Ren, 43 David's Road, Forest Hill, London SE23 3EP.

Printer Connection

Write-on-Line is the working name chosen for a new user group dedicated to printer users. Nick Godwin, previously organiser of the ZX Broad-sheet ZX-80/81 user group, would like to hear from anybody interested in joining and participating in a printer group.

Printers bring users a great deal of grief at times, so a skills exchange which concentrates on them could prove valuable.

Interested users should write with a SAF to Nick Godwin at 4 Hurkur Crescent, Eyemouth, Berwickshire TD14 5AP, telling him printer/s they use or in which are interested.

1989

**Babani
Electronics
Radio &
Computer
Books**

THE FIRST VALUE TECHNICAL BOOKS AVAILABLE

Books by Babani

The two latest publications from Bernard Babani (Publishing) Ltd. are *A Concise Advanced User's Guide to MS-DOS* by N. Kantaris and *More Advanced Midi Projects* by R.A. Penfold. Babani publishes several dozen books on general and specific subjects in electronics, audio, amateur radio, computing and electronic music. There are three current QL-specific titles on the list: *An Introduction to QL Machine Code* by R.A. and J.W. Penfold, *Into the QL Archive* and *Counting on QL Abacus*, both by J.W. Penfold.

Almost all Babani books cost less than £5, many of them half that price, and a free complete catalogue can be obtained by writing to Bernard Babani (Publishing) Ltd, The Grampians, Shepherds Bush Road, London W6 7NF with your name and address.

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in *Sinclair QL World*. Whether you want to ask for help with a technical problem, provide somebody

with the answer, or just sound off about something which bothers you, write to: Open Channel, Sinclair QL World, Greencoat House, Francis Street, London SW1 1DG.

Sunset

Looking for old QL titles once published by Sunshine Books. I went to my local library and it had two Sunshine Books publications, *The Working Sinclair QL* and *The Sinclair QDOS Competition*; the latter has a complete list of pokes to alter the system variables.

The commands to be included in a program to change from upper- to lower-case, and vice versa, is:

10 POKE__W 163976, 6000:
REM All keyboard entry
after this is upper-case

or

10 POKE 163976,0: REM. All
keyboard entry after this is
lower-case.

Both commands can be overridden by toggling the CAPS LOCK key.

Andy Hollis
Wallington,
Surrey.

Editor's comment: So far as I know, all the QL books published by Sunshine are out of print. For historical reasons, sometime before I joined QL World, I saw the last of the Sunshine computer volumes packed in cardboard boxes —

Editor's notebook

It took a long time, but at last we have a result in the QL Artist of the Year Competition 1988.

Congratulations to Mark Swift and Mark Knight, judged winner and runner-up after much debate. Congratulations also to everyone else who entered. We had plenty of fun looking through the entries. Thanks to Ron Massey, who took the screenshots, to Sector Software, which found us the prize, and to the judges.

We hope to be able to run another art competition before long and we will have the prizes in the bottom drawer at the start. It should not take so long to reach a conclusion.

Thank you to everybody concerned.

Last time I talked about the weather it was to the effect that summer was in short supply. There is no such complaint this year. I am going to Wales for an outdoor weekend, however, so expect a rain-check soon.

not many of them — bound for a remainder merchant. Remainder bookshops are an occasional source of out-of-print books but libraries are better. A local library will often be able to order a volume which is not in stock if you can give the title and/or author's name.

Index In

John Roberts might like to know that there is a program, *Indexer*, in the Quanta library, designed to index text files. I wrote it about two years ago and used it to index a 70,000-word book. It is fairly fast and you can make up an index with single words or groups of words.

Hugh Maill,
Frazier Street,
London SE1.

Hot paper

With regard to thermal paper for the Serial 8056 printer, Inmac — telephone 01-740 9540 and local branches — was advertising 8.5in. by 80in. rolls of thermal paper for £2.55 a roll — £2.50 for larger quantities.

T. Heath,
Henley Green,
Coventry.

Editor's comment. Check current price and availability before ordering.

New Budget

In the *Budget Planner*, *SQLW* July 1989, the 'staircase matrix' formula does not work as given. Please can you give the

correct formula and explain the algorithm?

E J Sargeant,
Long Eaton,
Notts.

Mike Lloyd replies: This is just one of the letters which pointed out two small but crucial typing errors affecting the *Budget Planner* spreadsheet. The comma before the final zero in the formula should in fact be a "less than" symbol and there should be two closing brackets after the zero. The correct formula for Cell 06 is:

$\text{int}(\text{index}(\text{col}()) - 13, 6) / 12 * (\text{row}() - \text{col}() + 9 + 12 * (\text{row}() - \text{col}() + 9 < 0))$

The objective of the formula is to calculate how much would be saved by putting aside a regular amount per month for a year before withdrawing it all to pay a particular month's bills. The amount is found by dividing the month's bills by twelve, rounding down to the nearest integer. To understand the rest of the formula, replace "col()" with the month in which the bills must be paid and "row()" with any month of the year for which a balance is required. Ignore the "+9" as this is an offset imposed by the position of the matrix in the spreadsheet. If a £120 bill was to be paid each April, for example, the balance in, say, October, would be:

$£10 * (10 - 4) = £60$

For months prior to April this formula produces a negative result which is corrected by adding twelve to the calculation. Thus, in February, the balance will be:

$£10 * (2 - 4 + 12) = £100$

I hope this explains the majority of readers' queries on this point. to everyone who struggled with the incorrect formula, my sincere apologies.

TROUBLE

Bryan Davies makes enquiries and offers advice on writing for The Progs.

At the time of writing, the Report Generator modules for *FlashBack* had not been completed and we cannot yet print the second part of the review started in the June issue. Letters of apology and explanation have been sent by the supplier to those who ordered *FlashBack* Special Edition. By now test copies of the new routines should be with the guinea pigs and it is hoped that the final part of the review will appear soon.

The revised *FlashBack* was completed several months ago. The *Lightning* "go faster" program is ready for release in its Special Edition form. It will be available on ROM or disc; the former will plug into the ROM port at the rear of the QL.

The program will be faster, of the order of 15-30 percent depending on machine and program configuration, and will allow improved scrolling. What will be interesting to investigate is the claimed ability to exert control over the screen printing in windows of other programs, including notoriously unco-operative ones such as Quill.

Media Manager

Another Special Edition should be available by the time this issue appears – *Media Manager*. The supplier admits that the original version of the program was not the most friendly piece of software. Its merits were that it enabled some recovery of corrupted QL files and transfers could be made of MS-DOS files to the QL. The new version is not a re-write, as experience with the original dictated so many changes that would have been pointless. It is new, and from the pen which has produced or been involved in the design of more good QL programs than any other it should take no guessing to figure who that is.

It is surprising how many programs are sent to *QL World* for consideration as Program of the Month, or for inclusion in Microdrive Exchange. What is even more surprising is the generally reasonable quality of them. The problem is what to do with them. If the magazine were to devote more space to printing programs, there could be complaints from readers who are

not interested in typing-in SuperBasic programs.

For a program to have a better-than-average chance of being selected, it needs to be short enough to fit into the space allocated to *The Progs*, should preferably be written in SuperBasic – with an assembler version also if desired – and, above all, it has to be of interest to a significant number of readers.

There is no point in producing the thousandth variant on the Space Hero theme unless the conception and execution are exceptionally good. Likewise, others have produced games like draughts, whist, bingo and so on. On the other hand, a monster project to produce the definitive front-end program for the QL is equally doomed; if you can really write such a program, it is better to contact one of the well-known software suppliers.

There are some companies which are worth knowing about because they cater to those who do not have deep pockets. Two are **Greenweld Electronic Components** and **J.N. Bull** – see Information panel. Both specialise in picking up discontinued products and selling them at low prices and the range is very wide – for example, hair dryers at two for £5, ni-cad battery chargers at two for £1. They usually have a selection of computer units, such as keyboards, and occasionally have QL-related items.

Bull was selling ICL OPD Microdrive units at £5 each – two drives – but I believe it is out of stock of them. Greenweld advertises OPD circuit boards at £5, complete with 68008 CPU chip, the ROM chips containing the operating system, and various other components; the 68008 on its own is offered for £3.

The OPD 68008 I tried appears to work normally in the QL – although no guarantee of function is given at this price level, naturally. Bull offers cartridges "for the double Microdrive", presumably a reference to the OPD units at £1.50 each or four for £5. Disc drives and their power supplies are relatively cheap.

CP/M formats

Maurice Richardson complained that the Success CP/M emulator would not convert his unspecified disc formats. CP/M formats vary considerably; there is nothing like the uniformity found in the MS-DOS world. The program was designed to run CP/M programs in certain formats directly and to convert programs in other formats into a form in which they could be run. The only format I know to be

unconvertible is that of the Amstrad PCW; that could be converted by the 1.0 version of the program but not by the later one.

Apart from the format in which data is written to the disc, there is the matter of the disc drive type; there is by no means complete compatibility between drive types and there are undoubtedly CP/M programs on discs which the QL drives will not accept. If you can find the instructions which arrived with your disc interface, look for a compatibility chart to determine whether or not the drive from which your CP/M discs came is compatible with the one you have on your QL.

Transform

Transform has largely moved out of the QL market but can still supply cartridges and cartridge boxes. It also provides support for Psion software. In the main, it now deals with the Psion Organiser and associated units and PCs, particularly in networks, but also handles Z-88 items. **Transform International** is a separate operation and still supplies its usual range of QL products.

There has been some discussion as to whether or not *Front Page Extra 2* is fully usable. A version 1.03 master disc was sent by **B. Hands**, with the information that printing with a copy made from it was unsuccessful, only a few lines of the document appearing on paper. With no corrective work being necessary, another copy made on my system produced reasonable printout.

M.C. Holland finds that he gets blank spaces in lines of text from version 1.05 for the basic QL, using a Citizen 120D printer. Again, a copy made from that produced reasonable print on my printer. Normally I would say the problem is in entering the printer parameters, in particular those which determine the vertical spacing increments, for instance the ESC "A" + n sequence commonly used to set 1/72in. line spacing.

Another place to make a mistake is when asked how many codes your printer requires for line feeds; as the answer can only be one or two, it does not take long to try both possibilities.

What seems to be a standard question in PC programs also is what setting has been made with printer DIP switches for line feeding – auto or manual. Again, there are only two possibilities and trying both should not take long. Unfortunately, in Holland's case at least, none of the questions had to be answered, because the printer concerned was Epson-

SHOOTER

E M S O L V E D

compatible and indicating that during the set-up process was about all the user was allowed to do. The only other areas which seemed likely to produce bad printing were incorrect setting of DIP switches in the printer and some fault in the inter-connecting cabling. The latter point appeared at first to be a non-starter, since printing from Quill had been satisfactory on the unexpanded QL.

Investigation with the printer and serial-parallel interface concerned revealed that some of the printer DIP switch settings needed changing but that did not fix the basic problem, which showed as horizontal gaps in lines of printed text. At the 0,0 character size there was a repeated pattern of form lines, the first of which was printed correctly, the other three suffering from missing or misplaced sections. The problem was fixed by substituting another serial-parallel interface.

Although standard text printing, from

Quill and *text87*, was satisfactory, graphics printing was not. It remains to be established whether the fault is inside the interface or in the wiring to an from it. The wiring is a possibility because the particular interface concerned has a 9-pin "D" connector, for use on QLs made for overseas markets such as Germany, and there seems to be some uncertainty

concerning the wiring to the serial port connectors in those machines.

Comments from users of Front Page for unexpanded and expanded QLs would be welcome; in the continued absence from the QL scene of the author of the program, it may be possible to pool experience and help some users get their printers working properly.

INFORMATION Media Manager Special Edition, Lightning Special Edition:

Digital Precision,
22, The Avenue,
Chingford,
London E4 9SE.
Tel: 01-527 5493

FlashBack:

Sector Software,
39 Wray Crescent,
Ulmes Walton,

Leyland,
Lancs PR5 3NA.
Tel: 0772 454328.

Greenweld Electronic Components,
443 Millbrook Road,
Southampton SD1 0XH.
Tel: 0703 772501/783740

J & N Bull Electrical,
Dept. MM
250 Portland Road,
Hove, Sussex BN3 5QT.
Tel: 0273 734648/203500.

TECHNIQL

A two-dimensional CAD package suitable for all general, scientific and engineering applications

Create accurate, finely detailed plans, diagrams or designs.

- * Zoom in and out.
- * Library of drawing tools
- * Fast, multi-width output
- * 2 screen modes

£49.95

DISK VERSION
NOW AVAILABLE

CARTRIDGE DOCTOR

Essential for all QL owners.

- * Rescue files from damaged cartridges
- * Recover newly deleted files
- * Recover files with damaged blocks
- * And much more

£17.95

ASSEMBLER WORKBENCH

A complete set of tools for the machine code programmer.

Combines assembler, monitor and screen editor. Dual screen to assist debugging of graphics programs, can operate on RAM or disc files. Compact and easy to use.

£24.95

STRIP POKER

Challenge the luscious Denise to a riveting game of cards.



£14.95

COSMOS

Identify 500+ stars and planets with this impressive astronomy program.

COSMOS displays accurate star maps for any date and time anywhere in the world. View the solar system, the moons of Jupiter, Saturn's rings - any visible object in the sky

£14.95

DISK VERSION
NOW AVAILABLE

DEATHSTRIKE

An exciting 'Scramble' game.

Maneuver your ship through alien territory, gain points by hitting targets with bombs and missiles. Your final objective is to destroy the mothership with an accurately placed bomb. A fast addictive game with excellent graphics.

£17.95

TALENT+

Stone Street, North Stanford, Ashford,
Kent TN25 6DF.
Tel: 0303 813883 Fax: 0303 812892 Telex: 966676 PMFAB G



Ron Massey gets to grips again with Quill and printers.

• PSION •

Jacob Lewin would like to be able to number Quill pages in excess of its maximum 254-page starting limit. This subject has been covered several times in Psion solutions. Quill will not allow you to start your page numbering beyond 254 but will continue to count until you run out of memory.

There are three ways to circumvent this problem, two of which involve using separate programs. The first solution requires two passes through the printer; the second only one. Neither of the first two is an elegant solution but will get the job done.

Using listing one, print a document in two stages; first print the document with the upper and bottom margins set so that the block of text appears in the middle of the page. Re-load the paper into the printer – tractor feed paper is best for this system. A second pass is used to number the pages and add any headers/footers at the same time.

This simple program can be modified to suit almost any supplementary printing required but will probably need experiment to get the header, footer and paper positioning correct. The CHR\$ codes are for Epson-compatible printers; if your printer requires different codes, substitute them where appropriate.

Line 110 sets the page length formatting to 66 lines with CHR\$(66). Line 120 sets the page number counter to "1". Line 140 is a pause and can be deleted if you have faith that you will not suffer from paper jamming.

Line 150 prints the header, followed by a Form Feed. Paper length in line 110 should be set so that the paper is positioned to the first line of the footer, beneath the block of text.

Line 170 is the footer, followed by the current page number. The procedure, Set_line_spacing is a FOR loop and should be set so that the paper is positioned at the top of the next page. This loop allows

four lines between the last line of the footer and the beginning of the next page and its header.

The last line in the repeat loop increments the counter and returns to the beginning of the loop and pauses until a key is pressed.

Listing two will print a __lis file, adding headers and foot-

very large documents is to use *The Editor* and its companion utility, *EDT_PRT*.

If you have produced the document in Quill, print the document to a file and load it into *The Editor* with the RD command. This will provide page breaks and all that is required is that you run

produce ASCII files?". The answer, in a word, is no. As proof of this, type-in listing three.

This program will produce two files. Test1__lis is the entire table of ASCII characters beginning with a CHR\$ number, followed by two spaces and the character. Test2__lis is identical, except that it begins the CHR\$(29) – CONTROL SHIFT].

Start Quill, press <F3> <0> <F> <I>. You will then be asked to type-in the name of a file to import. Load each of the two files in turn into Quill. The first file will load to CHR\$(13), perform a forced page break and continue importing up CHR\$(26) and then abort the import at CHR\$(27), the ASCII abort code also used as the QL ESCAPE key.

Test2__lis will begin with CHR\$(29) and import all the character codes Quill supports. Characters upwards from 192 will not be displayed in Quill but will be shown as blank spaces. So far as Quill is concerned the ASCII characters do not exist.

A document printed to a file without a printer driver includes Quill codes for character changes. If you have a printer driver available the codes generated by Quill are converted to whatever definitions you have made in your printer driver.

The content of a __lis file depends on your printer driver – or solely on Quill if you do not have a printer driver available. An example of the type of file produced by my driver is as follows.

The codes shown here as CHR\$ numbers are the characters entered into a __lis file as characters. They are shown here as codes because neither Quill in which this is being written, nor printers, will reproduce the characters required. Characters shown in quotes are the characters required:

27 "a" 27 "I" 1 27 "M" This is an Elite line. 13
27 "G" This is a Bold line. 27
"H" 13
27-1 This is an underlined line.

Listing 1: printer driver 1.

```
100 OPEN#4,ser1
110 PRINT#4,CHR$(27);"C";CHR$(66)
120 n = 1
130 REPEAT loop
140 a$ = INKEY$(-1) : REMark A pause for each page.
150 PRINT#4,to 20, 'This is a header.'
160 PRINT#4, CHR$(12) : REMark Form Feed
170 PRINT#4,to 10, "This is a footer on page "; n
180 Set_line_spacing
190 n = n+1
200 END REPEAT loop
210 :
220 DEFINE PROCEDURE Set_line_spacing
230 FOR i = 1 to 4
240 PRINT#4
250 END FOR i
260 END DEFINE
```

Listing 2: character generator for importing into Quill.

```
1 OPEN_NEW#3,mdv1_Test1_lis
2 FOR i=0 to 255
3 PRINT#3,i!!CHR$(i),
4 END FOR i
5 CLOSE#3
6 OPEN_NEW#3,mdv1_Test2_lis
7 FOR i=29 to 255
8 PRINT#3,i!!CHR$(i)
9 END FOR i
10 CLOSE#3
```

ers where appropriate. This program requires that you do three things:

1. Use Quill as you would normally, including any font and translate changes.
2. Format your Quill document to "0" page length.
3. Set the footer and header to "NONE".

There is only one mandatory input required by this program, the number of lines of text per page. Other than this the page position from the footer to the top of the next page is determined by the number of lines moved by the form feed in line 24.

If you have the program available the best and third alternative for page numbering

EDT_PRT. In addition to page numbering, this method provides you with as many translate characters as the QL and your printer are capable of producing.

A query has arisen from a comment made by Peter Forbes in the July, 1988 Open Channel with regard to Quill embedded (hidden) printer control codes.

The letter says: "He mentions spurious characters added to the imported text. Since removing the cartridge containing the printer driver cures this, presumably they are added, as control codes, by the printer driver when the text is 'printed' to a file. Could this be a way of persuading Quill to

SOLUTIONS.

27-0 13

27 P 27 15 This is a condensed/superscript line. 18 27 "M" 13
27 4 This is an italics/subscript line. 27 "S" 13

The above prints as:

This is an Elite line.

This is a Bold line.

This is a condensed/superscript line.

This is an italics/subscript line.

The group of the first seven characters in this file is the printer preamble code. The first line of text is offset in the file by the length of the first group but, because the printer recognises the group as control codes, printed spacing is as seen in Quill. The last character on each line is the CHR\$(13) end of line/newline code.

All the control codes at the beginning and end of each line are embedded codes, indicated in Quill only by a change of typeface and/or colour. Once

you have printed a document to a file, because they contain all of the information required by a printer, data from Quill _lis files can be sent directly to a printer by typing:

**COPY _N drive_filename-
_lis TO seri**

The COPY_N command performs the same function as a COPY command, except that the former strips off file header information. Both commands have the same effect with JS QLS but can produce spurious characters from JM or AH machines if the COPY file to ser is used.

Another QL enthusiast writes to say that he has encountered several Quill-related problems - irrecoverable def_tmp files, importing a _lis file, and confused Microdrives.

In order, the Quill mechanism for enabling users to produce documents larger than the

Listing 3: A slightly more sophisticated header and footer printer.

```
1 OPEN_IN#3, ram1_temp_lis
2 OPEN#4,ser1
3 REMark Set length of Form Feed
4 REMark CHR$(5) = 5 lines to bring paper to
5 REMark the top of the next page.
6 n=1 : pgnum = 1 : flg1 = 0 : wdlin = 96
7 REPEAT in_loop
8 INPUT "Number of text lines per page : ";lnum$
9 IF lnum$ = "" : CLS : NEXT in_loop
10 INPUT "Text to be printed in header : ";header$
11 IF header$ = "" : header$=" "
12 INPUT "Text to be printed in footer : ";footer$
13 IF footer$="" : footer$=" "
14 EXIT in_loop
15 END REPEAT in_loop
16 Position_text
17 REPEAT prt_loop
18 IF EOF(#3) : CLOSE#3 : EXIT prt_loop
19 INPUT#3, a$
20 IF flg1=0 : PRINT#4,TO pos1,header$
21 PRINT#4\\
22 flg1 = 1
23 END IF
24 PRINT#4, CHR$(27);"C";CHR$(6);
25 PRINT#4, a$;
26 n = n+1
27 IF n = lnum$
28 n = 1
29 PRINT#4,\\ TO pos2, footer$;" " : pgnum
30 pgnum = pgnum+1 : flg1 = 0
31 PRINT#4,CHR$(12);
32 END IF
33 END REPEAT prt_loop
34 PRINT#4,TO pos2,footer$
35 PRINT "End of File"
36 :
37 DEFINE PROCEDURE Position_text
38 REMark Elite = 96 chars wide
39 REMark Pica = 80 chars wide
40 pos1 = INT((wdlin/2)-(LEN(header$)/2))-2;
41 pos2 = INT((wdlin/2)-INT(LEN(footer$)/2)-LEN(pgnum)-3)
42 END DEFINE
```

Printed with this utility program. Quill's page length set to 0,
Upper margin set to 0, footer and header to None.

**** HEADER PRINTED WITH THIS UTILITY ****

```
1 OPEN_IN#3, ram1_temp_lis
2 OPEN#4,ser1
3 REMark Set length of Form Feed
4 REMark CHR$(5) = 5 lines to bring paper to
5 REMark the top of the next page.
6 n=1 : pgnum = 1 : flg1 = 0 : wdlin = 96
7 REPEAT in_loop
8 INPUT "Number of text lines per page : ";lnum$
```

A footer for an 8-line document. Page: 1

**** HEADER PRINTED WITH THIS UTILITY ****

```
9 IF lnum$ = "" : CLS : NEXT in_loop
10 INPUT "Text to be printed in header : ";header$
11 IF header$ = "" : header$=" "
12 INPUT "Text to be printed in footer : ";footer$
13 IF footer$="" : footer$=" "
14 EXIT in_loop
15 END REPEAT in_loop
16 Position_text
```

A footer for an 8-line document. Page: 2

**** HEADER PRINTED WITH THIS UTILITY ****

```
17 REPEAT prt_loop
18 IF EOF(#3) : CLOSE#3 : EXIT prt_loop
19 INPUT#3, a$
20 IF flg1=0 : PRINT#4,TO pos1,header$
21 PRINT#4\\
22 flg1 = 1
23 END IF
24 PRINT#4, CHR$(27);"C";CHR$(6);
```

A footer for an 8-line document. Page: 3

**** HEADER PRINTED WITH THIS UTILITY ****

```
25 PRINT#4, a$;
26 n = n+1
27 IF n = lnum$
28 n = 1
29 PRINT#4,\\ TO pos2, footer$;" " : pgnum
30 pgnum = pgnum+1 : flg1 = 0
31 PRINT#4,CHR$(12);
```

*Functionally,
a large
document
can be
compared to
an audio
cassette.*

available memory is to write a temporary file as a sort of scratch pad. As you move through the file for editing or whatever, Quill keeps track of your position relative to the temporary file and updates when necessary.

Functionally, a large document can be compared to an audio cassette. If you pull a length of tape from the cassette

• PSION • SOLUTIONS •

the free tape outside the cassette represents the file in memory and the remainder of the tape inside the cassette is the def_tmp file.

Maintaining the same length of free tape, as you wind the cassette in either direction the data on the free tape changes relative to either end of the remaining tape. The Quill mechanism for increasing the QL storage capacity for large documents effectively works very much in this way.

When you save a large document, Quill sorts out the memory and def_tmp resident parts of the file and assembles it into a continuous document. If there is insufficient memory, re-loading a document causes Quill to open another def_tmp file immediately.

Because of the structure of a def_tmp file and since it is not a legitimate Quill document, Quill will always refuse to load, import or merge it. Once Quill has lost its place in the file it is

usually irrecoverable by any means other than a file editor such as the Ark Spy or The Editor and a considerable amount of manual editing.

The second problem concerns importing a _lis file using M. Becket's printer driver for the 8056 which appeared in the September, 1986 issue of QL World. If a document is

not include characters which Quill recognises as program commands. See the spurious character problem.

No immediate answer to the final problem, confused Microdrives, springs to mind. This correspondent says that:

"On loading a document from one cartridge in mdv1_, then attempting to merge or

infrequently-used Microdrives, and the problem would not rear its head even once. I used the three loading commands, Load, Merge and Import – the latter for a _lis file of the document – for the appropriate file types, from different cartridges, all from mdv1_ and everything worked as expected.

If I had to make a rough guess I suggest that it is possible that, in sequence, this is what may have occurred. A _doc was loaded from cartridge one. An attempt to LOAD a _lis file may have been made from cartridge two, upsetting the Quill balance by the attempt. The writer then attempted to LOAD another _doc file, pushing Quill over the edge, so to speak.

I would be interested in hearing from other readers who may have experienced this problem. Please include a detailed account of the sequence of events.

No immediate answer to the final problem, confused Microdrives, springs to mind.

printed to a file using the FX80 driver, the document can be imported into Quill. Quill will not import, load or merge a _lis document if the 8056 driver is used.

First, the load or merge commands require a Quill _doc and its attendant format. Import, however, will work with any ASCII file which does

import another document from a different cartridge, also in mdv1_. On returning to the original cartridge, an error report "Corrupt" is written to the header of the document. This has the effect of rendering the file inaccessible from Quill."

I tried duplicating the problem with several document types, after dusting my

QL SUPERTOOLKIT II by Tony Tebby THE ULTIMATE QL ENHANCEMENT

Over 118 Commands:— Full Screen Editor, Key Define Print Using, Last Line Recall, Altkey, Job Control, File Handling, Default Directories, Extended Network.

16k Eprom Cartridge Version (r) £ 24.15d
Configurable Version on Microdrive (r) £ 23.00d

MIRACLE SYSTEMS PRODUCTS

QL Expanderam Ok board, fit your own drams. £ 23.00d
QL Trump Card 768k (Toolkit II etc) (r) £259.90b
QL Trump Card 512k (Toolkit II etc) (r) £213.90b
QL Trump Card 256k (Toolkit II etc) (r) £149.50b
QL Expanderam 512k Thru Card £133.40b
QL Expanderam 256k Thru Card (r) £ 80.50b
Dra..ns to suit above 41256/15 (r) £ 8.05c
QL Modem (r) £ 48.30d
QL Centronics Printer Interface (r) £ 28.75d

QL HARDWARE

Single 3.5" Disc Drive & (Own PSU) (r) £ 97.75a
Dual 3.5" Disc Drive & (Own PSU) (r) £188.60a
Q POWER REG. The only real solution to your QL overheating
(switched mode power supply run cold) (r) £ 23.00c
QL Keyboard Membrane (r) £ 11.50d
QEP III Advanced Eprom Programmer (r) £121.90d
Care Eprom Cartridges each (r) £ 5.75c
Eprom 27128 250n/s 16k (r) £ 5.75c
ULA Chip ZX8301 (r) £ 15.64c

MAGNETIC MEDIA

Microdrives (each) (r) £ 1.98c
3.5" (each) d/s disc (r) £ 1.61c
3.5" (10 of) d/s discs (r) £ 13.80c

SOFTWARE 87

TEXT 87 V. 2.00 (r) £ 45.00d
FOUNTEXT 89 (r) £ 15.00c
FOUNTEXT 88 (r) £ 25.00c
TEXT 87/FOUNTEXT 89/FOUNTEXT 88 (r) £ 79.00b
2488 PRINTER DRIVER (r) £ 15.00c

HOW TO ORDER:

ALL PRICES INCLUDE VAT

By Post. Enclose your Cheque/PO made payable to CARE Electronics.
Or use ACCESS/VISA. Allow 7 days for delivery

TONY TEBBY SOFTWARE (QJUMP)

QPAC II new from the house of QJump, a totally new version of QRAM and QPAC. Available September 89.

Please phone for further details.
QFLP (Micro/P disc interface upgrade) (a) £ 14.95d
QMON II Microdrive (a) £ 19.95d
QMED (Medic disc interface upgrade) (a) £ 14.95d
QPTR Pointer Interface m/drive (a) £ 34.50d
QPTR Pointer Interface + 3.5" disc (a) £ 29.90d
QTPY Type/Spell Checker (a) £ 29.90d

ZITASOFT SOFTWARE by Steve Jones

LOCKSMITH copies M/DRIVE — M/DRIVE (r) £ 11.50c
4MATTER + LOCKSMITH copies M/DRIVE — DISC (r) £ 23.00c
The above programs are not for use in the UK.
SHRIVEL memory shrink prog user definable in 128k or 192k or 256k
etc (a) £ 6.90c
TOOLCHEST utilities to allow the creation of customised mdv doctor
prog (r) £ 11.50c

SIDEWINDER — High resolution printer driver prints full screens
or parts of screens from postage stamp size to large banners.
Prints sideways, invert, scale, mirror, text insertion. (r) £ 17.25c

SIDEWINDER PLUS — (for expanded QL's) includes all the
features of above, PLUS multiple label printing, desktop
publishing files and printer driver for 24" pin plus LC10 and 3x80
colour printers. (Please state 3.5" disc or M/D) (r) £ 23.00c
Upgrade to Sidewinder Plus (r) £ 8.05c

MONITORS (Price including lead)

Philips BM7502 Green Hi-Res (r) £ 89.93a
Philips CM8833 Colour Med-Res (r) £271.41a
Philips AV7300 TV/tuner for above (r) £ 69.00b

READYMADE LEADS

RGB QL to Phono (r) £ 5.75c
RGB 8-6 pin DIN (r) £ 7.13c
RGB 8-7 pin DIN (Hitachi) (r) £ 7.13c
RGB 8-7 pin DIN (Ferguson) (r) £ 7.13c
RGB 8 pin to SCART (Euro) (r) £ 11.50c
6-way PCC 25way 'D' (Printer-Ser 1) (r) £ 9.89c

**CARE
ELECTRONICS**
OPEN
9am-5pm Mon-Thu
9am-4pm Fri-Sat

800 ST ALBANS ROAD,
GARSTON, WATFORD,
HERTS. WD2-6NL.
Tel: 0923-672102

**Q POWER
REGULATOR
NEW
SIDEWINDER
Text⁸⁷
NOW IN
STOCK**

Please add carriage
a=£11.50 b=3.45
c=£1.38 d=£2.30

The information given in the January 19, 1989 *QL World* with regard to disc drives has settled many problems before they arose. There is, as always, some further information on this subject which may interest many readers contemplating this desirable type of expansion.

Prices for disc drives, especially the bare ones, are falling all the time, as well as disc prices. To benefit from this it is necessary to be somewhat enterprising in regard to the enclosure or case. If you can bend a piece of metal it is easy or you can buy a plastic case to house the drives. A simple hole needs cutting in the front, plus a hole for the wires. A power supply can be squeezed in, especially if the transformer is housed in the wall plug.

Bare new 3.25in. drives vary from £15 for 80-track, single-sided to £50 for 80-track, double-sided. Cheaper ones, again new, are Sony drives for £5 each. The snag with them are that they have only a 24-pin connector without the information on what pin does what.

Steer clear of the 1.6MB high-density drives unless you have very detailed knowledge. They look the same, have the same connections and appear to work when plugged in. On the ones I have tried, I get FORMAT FAILED. This is due to the motor running at a higher speed with the WD 1770 chip in the interface on the QL being unable to handle the data. A few QL interfaces from one manufacturer have a different chip, so the high-density drives work correctly. You cannot swap chips as there are 28 pins on the WD 1770/1772 and 40 on the other chip.

Hard disc systems are now starting to appear both from Miracle as a plug-in-and-go unit and from Rebel Electronics,

DISCS2

Dennis Briggs
has more to add
on discs,
including a case.

following what others have done before, calling them fold-over boards or some similar name. Some enterprising QL owners have even built their own from Veroboard back to back and on rare occasions they work.

To cure the crashes due to fitting the long dangly bits and at the same time prevent the QL crashing, put some metal round the expansion. It is not feasible just to wrap all the electronics in baking foil as it will short everything. It is easy to spray the inside of all plastic cases with a nickel screening spray. Use a meter to check that the metal layer is continuous; then, when dry, give it a coat of lacquer.

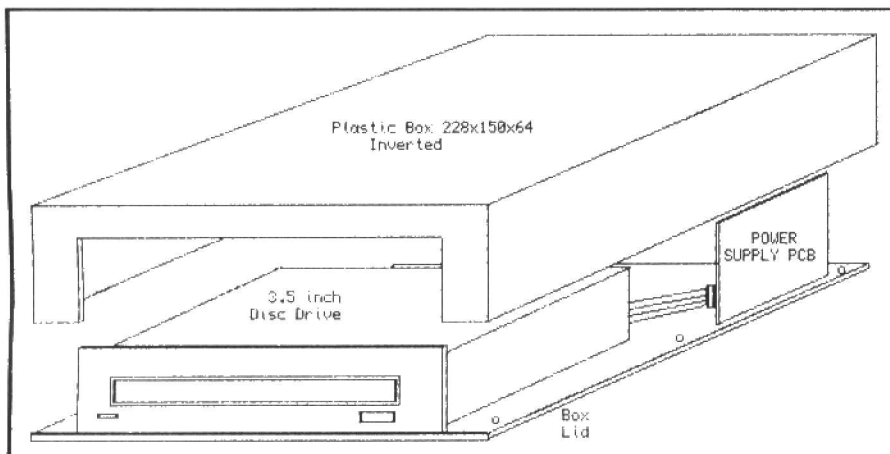
Now the easy part. Spray the back of the board with varnish and, when dry, stick self-adhesive copper foil to cover all the tracks. If in doubt put some Fablon over the tracks and then the foil. This puts a ground plane under the board. The much-maligned but still-loved Medic board had a ground plane in the middle layer. That is where it should be.

QL hardware-based crashes can also

they have been retrieved by using DISCED to look for what is on the disc with a little SuperBasic plus Toolkit 2 to get it back.

Lost Archive database files are much easier as I use the Quanta — free to members — program to close the file or the DP *The Editor* to make some sense of it. Use RU to read it in; then mark and delete the block at the top until the start of the first recognisable entry. Go to the bottom and do the same on all the ordering data. Then look at the records, find what is needed and what is rubbish then use the commands to get rid of the rubbish. Split the lines into suitable lengths, and again get Editor to take out what is not needed, such as the non-ASCII stuff. What you are aiming at is the field names, at the top in quotes, with a comma between, then each record on a separate line with a chr\$(10) at the end. If you do not understand, just export a _dbf

The problem with expansions is that the QL goes from Eurocard size to unwieldy.



A 3.5in. disc drive case.

as either a complete unit or parts. There are also a very few of the old CST hard disc interfaces, most of which have had minor or major surgery for one reason or another.

The main problem with all these expansions is that the QL goes from Eurocard size to something which is long and unwieldy. To overcome the problem, Rebel has introduced a backplane card

be reduced to nil by using HCT logic chips in the QL instead of the LS variety, putting a 4.7uF tantalum capacitor on every fourth QL RAM chip and connecting the ground of the 68008 CPU to the ground of the 8301 with a 0.1uF capacitor. An improved power supply then makes the QL rock solid.

I have lost files from disc and cartridge and also lost Archive files. In all cases

file to disc and read it back into Editor. Whatever you put in that format, Archive will swallow it if it has _lis on the end and is IMPORTED. It can even be persuaded to accept rubbish.

Example of a file suitable to be imported into Archive:

```
"Fname$", "Surname$", "street1$",  
"street2$", "Town$chr$(10)  
"Mary", "Freeman", "15 Wood Close",  
"Arman's Way", "Winsop"chr$(10)  
"Fred", "Bloggs", "1 Hampton Court",  
"Crab Boulevard", "Clingtown"chr$(10)  
"Jenny", "Smith", " " " " " "chr$(10)  
chr$(26)
```

Notice that there must be a record, even an empty one, for each field name.

One final point often overlooked is that if the Archive database file is re-ordered several times the total file length can become very large, as the ordering data is retained in the file. When you have the file in the order you wish, EXPORT it, perhaps to a RAM disc, and then import it back in. The order will be exactly the same as the original but the ordering data is removed, thus shortening the file.

PAPERWORK takes a Quantum Leap

*Graeme Young
lets Archive take
the strain of his
business
expenses and
weekly office
paperwork.*

Friday is paperwork day. I am a technical liaison officer for a major television manufacturer and travel a large part of eastern England from the Humber to the Orwell. At the end of each week and at certain times throughout the month my company expects returns which reflect my activity for the preceding period — expense claims, engineering reports, spares requisitions, itinerary, holiday forms and so on. Much of the stationery consists of forms which require filling in at the designated spaces by the sender and memos and headed letter paper. Some of the small administrative forms are usually produced on a duplicator which I have mimicked so that blank paper is all I need to stock.

In the days before the QL took over I fought a lonely battle with an office-size typewriter and a large bottle of correction fluid. Even in my novice days with the QL I used Quill for everything and totalled my expenses in my head, always getting them wrong. It was not long before I decided the computer should take the strain. Afficionados of the QL, from whom I sought advice, all agreed that Archive would be much more suitable for my work and so I burned some midnight oil, pestered fellow Quanta members, and eventually produced some very workable programs in Archive to retrieve, store and print information.

Daisywheel

The printer I use is a Triumph-Adler TRD7020 daisywheel type which gives excellent clarity. I delved into the printer driver program and got it to print characters such as Ω , μ R and μ , which are essential in my engineering reports. I also had the computer hold my expenses, much to the relief of the accounts department.

The QL is a JM ROM type with Trump Card, twin 3.5in. disc drives and a Prism QL14 colour monitor, though I started, like everyone else, with a bare 128K and the Microdrives. The real-time clock circuit is modified to take battery back-up. This is 99 percent successful and goes awry only if the computer hangs up for some reason. The entire system is mounted on a workbench from MFI; cables are tidied along the back of the bench.



The system — QL, Trump Card, twin drives, printer, monitor.

The printer is mounted on a small platform made of Contiboard to allow cables to pass underneath to the QL. Another platform raises the monitor slightly and permits reference books, papers and notebooks to occupy the bench-top underneath, where they are tidy yet still handy. I have set the monitor to one side, on my right, as I find neckache and eyestrain are less with it in this position. Power units for the computer and disc drives are tucked in the knee-hole.

The QL was notorious for hang-ups because of the heat and other problems in the 5V regulator. One obvious answer is to provide a larger heatsink for the device and to that end I have fabricated a heatsink from some aluminium channelling. It runs the whole length of the computer beneath the rear edge and replaces the feet used to tilt the keyboard. I also found that the regulator was not fitted with by-pass capacitors and they have been added to suppress any tendency to oscillate.

When I added a Trump Card I found the regulator on that was not provided with a very efficient means of conducting the heat generated. To overcome the occasional hang-up when first used for lengthy periods I made a bottom cover for the Trump Card module which doubled as a heatsink for the regulator IC. The regulator was re-positioned underneath the printed board so that its metal spine could make intimate contact with the heatsink/

bottom cover. A coat of matt black paint improves the radiation efficiency and therefore helps to dissipate heat and the finish matches the rest of the Trump Card module.

Trump Card

Recently I learned that there is a modification for Trump Card which is successful in eliminating hang-ups and I shall be installing it as well. According to Miracle Systems, the eight 560-ohm resistors, mounted in a row alongside an integrated circuit in the top right-hand area of the TC circuit board, should be lowered in value to 330-ohms each. This can be done by bridging each 560-ohm resistor with an 820-ohm, if you do not fancy de-soldering the very fine tracks.

Finally I re-designed the power supply. The original Sinclair unit is probably not totally adequate once Trump Card is installed — it runs very hot. For about £20 and in a few hours I made a unit which is well-regulated, runs very cool and has eliminated hang-ups due to power supply problems.

With the hardware operating satisfactorily I began to change many of the files from Quill 2.3 to Archive 2.3. Monthly expenses seemed to benefit the most from the transfer and had the added attraction of being stored on disc in a form readily exportable to Easel and Abacus for some fancy processing if required. Most of the other conversions to Archive followed

"MONPRINT"

```

proc comment
print "SET PRINTER LINE SPACING TO 1/8TH INCH - DIL PIN 7 UP."
print
print "Enter month (1 to 12) for expenses: "; input mon
select month=mon
print
print "Comment 1: "; input Comment1$
print "Comment 2: "; input Comment2$
print "Comment 3: "; input Comment3$
print "Comment 4: "; input Comment4$
print "Comment 5: "; input Comment5$
print "Comment 6: "; input Comment6$
print "Comment 7: "; input Comment7$
print "Comment 8: "; input Comment8$
endproc

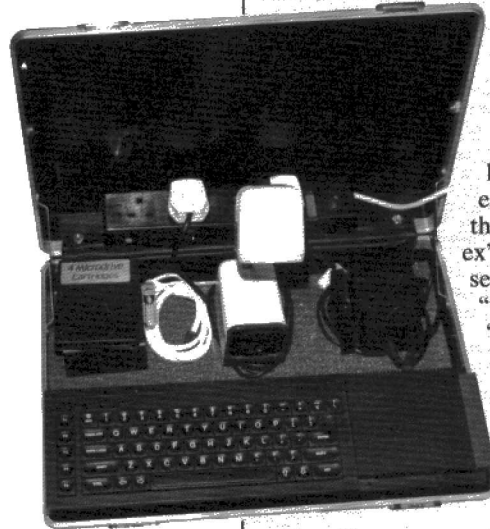
proc form
cls
comment
title
init
all
item
endall
hardco
display
endproc

proc hardco
let gap=18-(count()*2)
space;gap
let Gtot=Htot+TPtot+Etot+Ttot+Ptot+Stot+Mtot
let Gtot$=str(Gtot,0,2)
let Htot$=str(Htot,0,2)
let TPtot$=str(TPtot,0,2)
let Etot$=str(Etot,0,2)
let Ttot$=str(Ttot,0,2)
let Ptot$=str(Ptot,0,2)
let Stot$=str(Stot,0,2)
let Mtot$=str(Mtot,0,2)
lprint tab (46-L);Htot$; tab (67-L);Etot$; tab (78-L);TPtot$; tab
(89-L);Ttot$;tab (101-L);Ptot$; tab (111-L);Stot$; tab (122-L);
Mtot$; tab (133-L);Gtot$ space;10
lprint tab 5;"#1: ";Comment1$
lprint tab 5;"#2: ";Comment2$
lprint tab 5;"#3: ";Comment3$
lprint tab 5;"#4: ";Comment4$
lprint tab 5;"#5: ";Comment5$
lprint tab 5;"#6: ";Comment6$
lprint tab 5;"#7: ";Comment7$
lprint tab 5;"#8: ";Comment8$
endproc

let Htot=0
let Etot=0
let TPtot=0
let Ttot=0
let Ptot=0
let Stot=0
let Mtot=0
let Gtot=0
endproc

proc item
lprint tab 4;date$;
lprint tab 14;detail$;
let L=len(Amount$): let Amount=val(Amount$)
if Type$="H": let Htot=Htot+Amount: lprint tab (46-L);Amount$;
endif
if Type$="E": let Etot=Etot+Amount: lprint tab (67-L);Amount$;
endif
if Type$="TP": let TPtot=TPtot+Amount: lprint tab (78-L);Amount$;
endif
if $Type$="T": let Ttot=Ttot+Amount: lprint tab (89-L);Amount$;
endmf
if Type$="P": let Ptot=Ptot+Amount: lprint tab (101-L);Amount$;
endif

```



rapidly. Now the only files in Quill are my memos and company letters.

Fourteen database files resulted from the changeover and each has an attendant program. They are listed in a loose-leaf notebook for quick reference and show at a glance the file name, field names, program name and the procedures. Some of the procedures are used throughout the programs, such as 'sp' which does a line feed. The notebook is vital — my memory is not good.

Two of the programs work particularly well. Monthly expenses, which has the file name "monthex", and its attendant set of programs called "monprint", and "hotel" which was originally a database of hotels in which I could stay in my territory and which had conference facilities.

The latter has now grown into a large file with a set of programs for printing reservation confirmations for company and private use and the hotel address on an adhesive label or directly on the envelope. All addresses for printing on to labels and envelopes I have standardised to start at Tab 40.

Monthly expenses are submitted on a pre-printed form which has windows for name, number, department and columns and rows of boxes for the individual expense items. The layout is such that the bottom right-hand box contains the final amount which is the sum of the figures in the column above it and the row to the left of it. Each expense is entered into a record starting with the month — 1 to 12 — then the date of the expense, followed by brief details, the expense category, and the amount in decimal. When this is complete, the direct command <form is entered and this brings up a screen which asks for up to eight comments to be printed in the box provided on the expense sheet. It begins by asking for the number of the month, then the comments. When this is complete the printer starts. A warning is printed reminding me to set the printer to 1/8 in. line spacing.

With Hotels.dbf a similar thing is done. The hotel name, address, telephone number, and star rating is entered along with short comments about accommodation, food, cost of stay and what lecture/conference facilities there are. The program "hotprint" contains procedures for printing a confirmation of a company booking or a private one and for label/envelope printing. Programs one and two show the program listings and the field-names for the relevant database records.

For memos and letters I use Quill, which I find most acceptable now that I

run it with the DP *Lightning*. A template for each is stored on the disc containing the `__docs`. `Printer__dat` is copied to ram1 during boot-up; the Quill version used is configured specially for this and the saving in wear and tear on the drives is considerable. It is also faster. Once the memo or letter is written it is saved on to drive 2. The words of the prompt, "save flp2__memo__doc" are altered by adding a serial number after 'memo', so that it now reads "save flp2__memo45__doc". The same thing is done with letters and I keep a register of memo and letter numbers against their subject matter.

The TRD7020 printer will do graphics, though not from Easel, and the following program can be used to dump a screen, though I prefer now to use my old Serial 8056 which I keep connected and installed in the bottom drawer of the workbench. The Triumph-Adler will take about 20 minutes to complete a screen-dump, whereas the 8056 does the job in about three minutes. This process is very hard on the ribbon as it uses the full-stop character. When I used the daisywheel for produc-

```

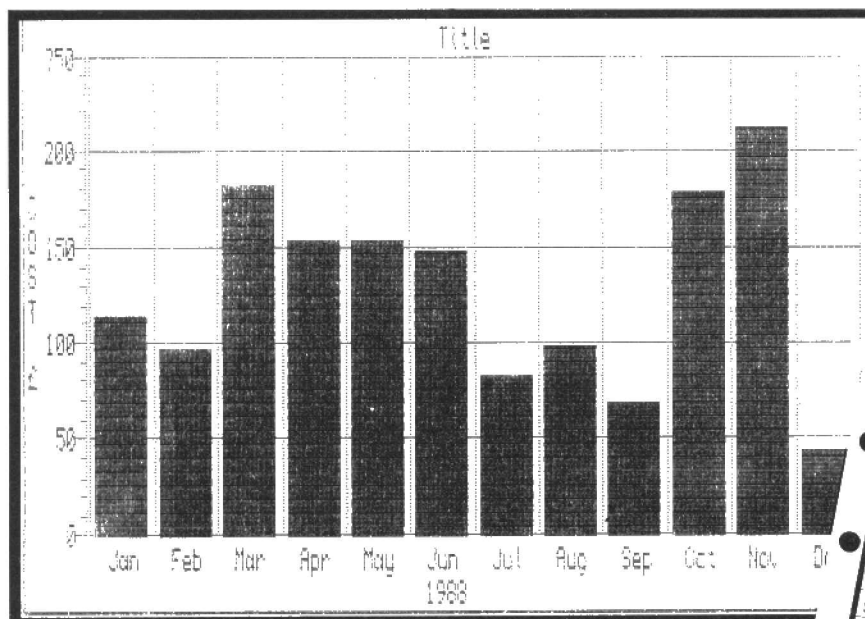
if Type$="S": let Stot=Stot+Amount: lprint taf$(111-L);Amount$;
endif
if Type$="M": let Mtot=Mtot+Amount: lprint tab (122-L);Amount$;
endif
lprint tab (133-L);Amount$
sp
endproc

proc sp
textpr;" "
endproc

proc space;y
let z=0
while z<y
sp
let z=z+1
endwhile
endproc

proc textpr;a$
lprint a$
endproc
proc title
lprint tab 53;"G.M.YOUNG."; tab 80;"TECHNICAL"
space;2
lprint tab 53;"200134"; tab 80;"3348"; tab 110;date(1)
space;4
endproc

```



ing graphics I remove the ribbon and use a sheet of non-smear carbon such as Silk Finish or NCR Ulti-max in front of the blank sheet. The printing process virtually destroys the carbon paper but one sheet of it is very much cheaper than a new ribbon. The procedure was first published by D. O. Nevill in *Popular Computing Weekly* for November 6-12, 1986; it assumes the use of a serial to parallel interface at 9,600 baud.

```

100 REMark **** SCREEN
    DUMP ****
110 REMark **** QL to TRD7020
    Printer ****
120 DEFINE PROCedure Dump__
    Screen

```

MONTHEX__dbf expenses database specification and export file.

```

130 LOCAL r$,b$,ll,r,c,w,b,z:BAUD
    9600: OPEN #3, ser1
140 r$=CHR$(27)&CHR$(51):b$=
    "":1=131070
150 FOR r=0 TO 255
160 PRINT #3,r$;CHR$(10);
170 FOR c=0 TO 127 STEP 2
180 1=1+2:w=PEEK(1)
190 IF w=0 THEN
200 PRINT #3,r$b$;
210 ELSE

```

```

220 FOR b=7 TO 0 STEP -1
230 PRINT #3,r$ "":z=
    INT(w/2/b)
240 IF z/2<>INT(z/2)
    THEN PRINT #3,r$
    "":
250 AND FOR b
260 END IF
270 END DEFINE c: END FOR T:
    CLOSE #3
280 END DEFINE Dump__Screen

```

MONTHEX__dbf
monprint__prg

MONTHLY EXPENSES DATABASE

Fields:

Month : Month 1 to 12 (Jan-Dec)

Date\$: Date DD.MM.YY

Detail\$: Brief description.

Type\$: Expense Category :-
H=Hotels E=Entertaining
T=Travel & Parking.
M=Miscellaneous.
S=Subsistence, Meals.

Amount\$: Amount in Decimal, NO £!

COMMAND : form. Prints expense form

Procedures:

comment : input screen for comment

form : runs all procs in seq.

hardco : prints totals + comments

init : sets all variables to 0

sp : prints an empty line.

space;y : prints y empty lines.

title : prints usual gen info

LISTING 2.

```

proc comcon
cls
print ;"CONFIRMATION OF HOTEL BOOKING (COMPANY)"
print "-----"
print ;"Date of Arrival?      : ": input Arr$
print ;"Date of Departure?    : ": input Dep$
print ;"Requirements?        : ": input Acc$
lprint tab 11;Name$
lprint tab 11;Add1$
lprint tab 11;Add2$
lprint tab 11;Add3$
lprint tab 11;Pcode$
space;1
lprint tab 63;date(1)
space;4
lprint tab 11;"Dear Sirs,"
space;1
lprint tab 11;"I have pleasure in confirming my recent reservation as
follows:-"
space;1
lprint tab 20;"Mr G.M.Young. Idiot's Lantern Limited."
lprint tab 20;"-----"
lprint tab 20;"Date of Arrival      : ";Arr$
lprint tab 20;"Date of Departure    : ";Dep$
lprint tab 20;"Type of Accommodation: ";Acc$
space;1
lprint tab 11;"For information, I am a Non-Smoker."
lprint tab 11;"I will be settling my account by ACCESS card, and in
looking forward"
lprint tab 11;"to my stay with you, I am"
space;2
lprint tab 11;"Yours Faithfully,"
space;4
lprint tab 11;"....."
lprint tab 19;"G.M.Young."
lprint tab 11;"Technical Liaison Officer."
display
endproc

```

```

-----
proc con
cls
print ;"HOTEL BOOKING CONFIRMATION"
print ;"-----"
print ;"Date of Arrival? : ": input Arr$
print ;"Date of Departure? : ": input Dep$
print ;"Requirements? : ": input Acc$
lprint tab 15;Name$; tab 74;"35, Fishbone Road,"
lprint tab 15;Add1$; tab 74;"Haddockshead,"
lprint tab 15;Add2$; tab 74;"Nibblingham,"
lprint tab 15;Add3$; tab 78;"NB19 5ZE."
lprint tab 20;Pcode$
space;3
lprint tab 74;date(1)
space;2
lprint tab 15;"Dear Sirs,"
space;1
lprint tab 15;"I have pleasure in confirming our recent reservation as
follows:-"
space;1
lprint tab 20;"Mr and Mrs G.M.Young."
lprint tab 20;"-----"
lprint tab 20;"Date of Arrival      : ";Arr$
lprint tab 20;"Date of Departure    : ";Dep$
lprint tab 20;"Type of Accommodation : ";Acc$
space;1
lprint tab 15;"For information, my Wife and I are non-smokers."
lprint tab 15;"I will be settling our account by ACCESS card. In looking
forward to "
lprint tab 15;"our stay with you,I am,"
space;2
lprint tab 15;"Yours Faithfully,"
space;5
lprint tab 15;"....."
lprint tab 21;"G.M.Young."
endproc
-----

```

For normal everyday correspondence I use a multi-strike fabric ribbon which has a long life, good clarity, and can be re-linked. If I know a document will be photocopied for distribution I put in the carbon ribbon; the quality of print from this is superlative and it photocopies very well. With output from my other printers, the Series 8056 and the Brother EP44, on thermal paper I take the precaution of photocopying anything I want to keep.

Thermal print fades in the light, so originals are always kept in the dark. Clear plastic document wallets also react with thermal and photocopy paper, so that I never use them unless the top page is covered by a blank sheet of copy paper. Easel will print only to the Serial 8056; the printer driver program is in listing three.

With Trump Card installed, and Taskmaster plus Lightning running, paper-work truly has made a quantum leap. I find Lightning an amazing utility which has completely removed any need I once felt for something better than Quill. The increase in speed and the ability to swap from one task to another, or into Super-Basic, is unimaginably handy. I have one caveat; I never invoke the Trump Card toolkit as I find it makes the system prone to crashes, particularly from inadvertent keyboard errors.

Toolkit

It could be argued that TK2 is not strictly necessary when running the Psion suite. The toolkit comes into its own when running the Psion suite. The toolkit comes into its own when the QL is engaged on other work unconnected with the weekly returns and the Psion four. It can always be called from Basic if one fancies living dangerously, but close all open files first. The occasional hang-up persists, usually associated with cursor movements or the Quill command F3 P and so on. I treat cursors and print commands with respect.

Frequently I work away from easy reach of home and stay in one of the hotels I have booked via the Archive program "hotprint". For those occasions I have fitted a JS QL into a briefcase. Also included in the case are a modified power supply, a small desk lamp and a four-way mains 13A socket block. I carry a 12in. monochrome TV, converted for monitor use, separately. At present I am using the Microdrives on this portable machine; they are surprisingly reliable provided they are kept clean and otherwise left well alone. The JS has the same regulator modifications as the JM and uses a modified QL UK2000 power unit. This machine is unexpanded but is adequate for most purposes and I intend to try Taskforce sooner or later.

The briefcase QL provides a useful portable office and it keeps down the hotel bar bills as well. As soon as my finances have recovered from the purchase of Trump Card I shall add disc drives and a little more memory to the outfit as there is


```

proc Hotadd
lprint tab 40;Name$
lprint tab 40;Add1$
lprint tab 40;Add2$
lprint tab 40;Add3$
lprint tab 45;Pcode$
endproc

```

```

-----
proc info
lprint tab 40;Name$;" ";Star$
lprint tab 40;Add1$
lprint tab 40;Add2$
lprint tab 40;Add3$
lprint tab 40;Pcode$
lprint
lprint tab 40;"Phone: ";Phone$
endproc

```

```

proc sp
textpr;" "
endproc

```

```

proc space;y
let z=0
while z<y
sp
let z=z+1

```

```

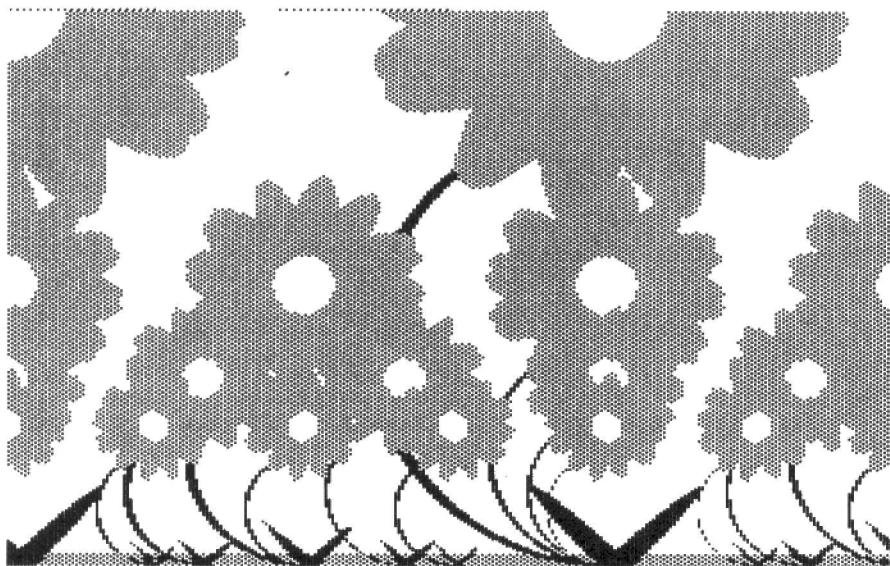
endwhile
endproc

```

```

proc textpr;A$
lprint A$
endproc

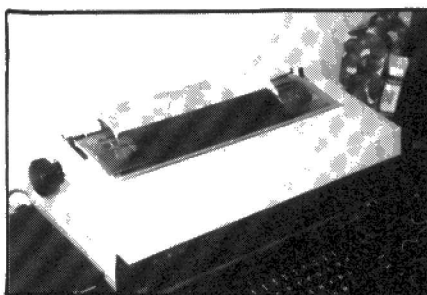
```



Above and below: a printout using carbon paper instead of ribbon.

plenty of room in the case. Until then, work done on this machine is transferred to disc when my home office is reached and any printing done where necessary.

I have a Brother EP44 thermal typewriter/printer which can be used with either QL. The supply of thermal paper for this and the Serial 8056 was beginning to dry up but the advent of fax machines has reversed the trend and now provides a plentiful supply of cheap thermal paper in



rolls. For high-quality printing the Brother paper in rolls or cut sheet is still the best, however, and I buy it whenever I see it on sale.

The EP44 will print on smooth plain paper using a thermal transfer ribbon costing about £2.75. The quality of the print is excellent but the ribbon does not last very long and not much use is made of this feature. It is only a pity that the EP44 will not do graphics, or so I am told. Does anyone know better?

With a smile

Fridays pass with a smile now, the drudgery of the old tyrannical typewriter transformed by the QL into speedy, efficient and pleasurable moments at the keyboard. Gone are the ranks of box files disfiguring my bookshelves and the reams of paper sometimes never disturbed twice. A year's work goes on three discs; another three are for back-up, kept in a neat Perspex box in the workbench. Here are two QLs earning their keep and giving great satisfaction, inasmuch as they do an excellent, accurate job quickly. Henceforth, fountain pens are only for signatures and paper is for feeding into printers.

Listing 3.

```

=====
100 COPY MDV1_GPRINT_PRT TO MDV1_FX80_PRT
110 A=RESPR(512)
120 LBYTES MDV1_GPRINT_PRT,A
130 DELETE MDV1_GPRINT_PRT
140 POKE_W A+400,6960
150 POKE_W A+402,0
160 POKE_W A+404,6987
170 POKE_W A+406,57345
180 POKE A+181,50
190 POKE_W A+474,33280
200 POKE_W A+496,7
210 POKE_W A+500,5
220 SBYTES MDV1_GPRINT_PRT,A,512

```

It is necessary either to start up Easel with the command BAUD 1200:LRUN MDV1_BOOT, or to incorporate the Baud 1,200 into the existing BOOT program. If you have two printers with differing baud rates and use Taskmaster, omit the lesser-used BAUD command. It can be inserted when the need arises by slipping into SuperBasic for a few seconds, typing "BAUD 1200" and returning to the current program. Remember to restore the commonly-used baud rate when you finish that task and proceed to another.

ARCHIVE REVEALED

Dennis Briggs unveils extra features and graphics characters hidden inside.

Hidden in the depths of Archive are some 30 extra features plus another 10 graphics characters along with the familiar ASCII character set. All are accessible by printing a special character code to the screen and while it would be reasonable to PRINT chr(65) instead of PRINT "A" it is essential that this approach be made if all the facilities are to be utilised.

When chr\$(12) is sent by SuperBasic to a device such as a printer, the printer will look in its ROM and respond to the instruction by driving the stepper motor an exact number of steps. The motor then turns the platen roller the precise distance of a sheet of paper, thus presenting a fresh sheet or a form feed. In the same way other characters sent to the printer make the printer behave in specific ways.

Screen prints

Printing to the QL screen is somewhat similar, except that chr\$ below 31 and above 128 will print a splodge, as they are not interpreted by the QL ROM. To reproduce the form feed on-screen, 22 empty print statements in Basic move the screen 'paper' up 22 lines to give a new sheet of 'screen paper' or CLS does it by a machine code routine in the ROM. It needs 66 empty 'PRINTs' to the printer channel to move up the real paper by 66

Archive Screen Driver Codes

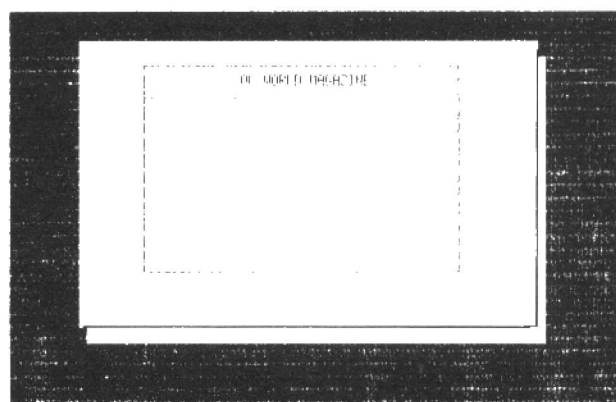
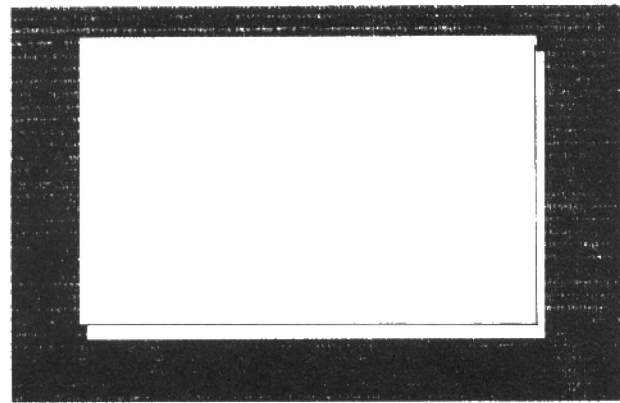
Driver code 0	No action	Parameters: 0
Driver code 1	Set ink colour	Parameters: 1
Driver code 2	Set paper colour	Parameters: 1
Driver code 3	_____ No action _____	
Driver code 4	Repeat characters	Parameters: 2
Driver code 5	Underline	Parameters: 0
Driver code 6	Cursor right	Parameters: 0
Driver code 7	_____ No action _____	
Driver code 8	Cursor left	Parameters: 0
Driver code 9	Tab	Parameters: 1
Driver code 10	Line feed	Parameters: 0
Driver code 11	Cursor up	Parameters: 0
Driver code 12	Form feed	Parameters: 0
Driver code 13	Carriage return	Parameters: 0
Driver code 14	Cursor on	Parameters: 0
Driver code 15	Cursor off	Parameters: 0
Driver code 16	_____ No action _____	
Driver code 17	_____ No action _____	
Driver code 18	Character plot type	Parameters: 1
Driver code 19	Delete character	Parameters: 0
Driver code 20	Define window	Parameters: 4
Driver code 21	Scroll screen up	Parameters: 1
Driver code 22	Scroll screen down	Parameters: 1
Driver code 23	Scroll left	Parameters: 1
Driver code 24	Scroll right	Parameters: 1
Driver code 25	Set boundary type	Parameters: 1
Driver code 26	Swap paper/ink colour	Parameters: 0
Driver code 27	ESCape sequences 'A' to 'H'	Parameters: 1
Driver code 28	CR+LF	Parameters: 0
Driver code 29	Pass through	Parameters: 1
Driver code 30	Home cursor	Parameters: 0
Driver code 31	Position cursor	Parameters: 2

```

100  print "aa"
101  print "bb"
102  print "cc"
103  print "dd"
104  print "ee"
105  print "ff"
106  print "gg"
107  print "hh"
108  print "ii"
109  print "jj"
110  print "kk"
111  print "ll"
112  print "mm"
113  print "nn"
114  print "oo"
115  print "pp"
116  print "qq"
117  print "rr"
118  print "ss"
119  print "tt"
120  print "uu"
121  print "vv"
122  print "ww"
123  print "xx"
124  print "yy"
125  print "zz"
126  print "aa"
127  print "bb"
128  print "cc"
129  print "dd"
130  print "ee"
131  print "ff"
132  print "gg"
133  print "hh"
134  print "ii"
135  print "jj"
136  print "kk"
137  print "ll"
138  print "mm"
139  print "nn"
140  print "oo"
141  print "pp"
142  print "qq"
143  print "rr"
144  print "ss"
145  print "tt"
146  print "uu"
147  print "vv"
148  print "ww"
149  print "xx"
150  print "yy"
151  print "zz"
152  print "aa"
153  print "bb"
154  print "cc"
155  print "dd"
156  print "ee"
157  print "ff"
158  print "gg"
159  print "hh"
160  print "ii"
161  print "jj"
162  print "kk"
163  print "ll"
164  print "mm"
165  print "nn"
166  print "oo"
167  print "pp"
168  print "qq"
169  print "rr"
170  print "ss"
171  print "tt"
172  print "uu"
173  print "vv"
174  print "ww"
175  print "xx"
176  print "yy"
177  print "zz"
178  print "aa"
179  print "bb"
180  print "cc"
181  print "dd"
182  print "ee"
183  print "ff"
184  print "gg"
185  print "hh"
186  print "ii"
187  print "jj"
188  print "kk"
189  print "ll"
190  print "mm"
191  print "nn"
192  print "oo"
193  print "pp"
194  print "qq"
195  print "rr"
196  print "ss"
197  print "tt"
198  print "uu"
199  print "vv"
200  print "ww"
201  print "xx"
202  print "yy"
203  print "zz"
204  print "aa"
205  print "bb"
206  print "cc"
207  print "dd"
208  print "ee"
209  print "ff"
210  print "gg"
211  print "hh"
212  print "ii"
213  print "jj"
214  print "kk"
215  print "ll"
216  print "mm"
217  print "nn"
218  print "oo"
219  print "pp"
220  print "qq"
221  print "rr"
222  print "ss"
223  print "tt"
224  print "uu"
225  print "vv"
226  print "ww"
227  print "xx"
228  print "yy"
229  print "zz"
230  print "aa"
231  print "bb"
232  print "cc"
233  print "dd"
234  print "ee"
235  print "ff"
236  print "gg"
237  print "hh"
238  print "ii"
239  print "jj"
240  print "kk"
241  print "ll"
242  print "mm"
243  print "nn"
244  print "oo"
245  print "pp"
246  print "qq"
247  print "rr"
248  print "ss"
249  print "tt"
250  print "uu"
251  print "vv"
252  print "ww"
253  print "xx"
254  print "yy"
255  print "zz"
256  print "aa"
257  print "bb"
258  print "cc"
259  print "dd"
260  print "ee"
261  print "ff"
262  print "gg"
263  print "hh"
264  print "ii"
265  print "jj"
266  print "kk"
267  print "ll"
268  print "mm"
269  print "nn"
270  print "oo"
271  print "pp"
272  print "qq"
273  print "rr"
274  print "ss"
275  print "tt"
276  print "uu"
277  print "vv"
278  print "ww"
279  print "xx"
280  print "yy"
281  print "zz"
282  print "aa"
283  print "bb"
284  print "cc"
285  print "dd"
286  print "ee"
287  print "ff"
288  print "gg"
289  print "hh"
290  print "ii"
291  print "jj"
292  print "kk"
293  print "ll"
294  print "mm"
295  print "nn"
296  print "oo"
297  print "pp"
298  print "qq"
299  print "rr"
300  print "ss"
301  print "tt"
302  print "uu"
303  print "vv"
304  print "ww"
305  print "xx"
306  print "yy"
307  print "zz"
308  print "aa"
309  print "bb"
310  print "cc"
311  print "dd"
312  print "ee"
313  print "ff"
314  print "gg"
315  print "hh"
316  print "ii"
317  print "jj"
318  print "kk"
319  print "ll"
320  print "mm"
321  print "nn"
322  print "oo"
323  print "pp"
324  print "qq"
325  print "rr"
326  print "ss"
327  print "tt"
328  print "uu"
329  print "vv"
330  print "ww"
331  print "xx"
332  print "yy"
333  print "zz"
334  print "aa"
335  print "bb"
336  print "cc"
337  print "dd"
338  print "ee"
339  print "ff"
340  print "gg"
341  print "hh"
342  print "ii"
343  print "jj"
344  print "kk"
345  print "ll"
346  print "mm"
347  print "nn"
348  print "oo"
349  print "pp"
350  print "qq"
351  print "rr"
352  print "ss"
353  print "tt"
354  print "uu"
355  print "vv"
356  print "ww"
357  print "xx"
358  print "yy"
359  print "zz"
360  print "aa"
361  print "bb"
362  print "cc"
363  print "dd"
364  print "ee"
365  print "ff"
366  print "gg"
367  print "hh"
368  print "ii"
369  print "jj"
370  print "kk"
371  print "ll"
372  print "mm"
373  print "nn"
374  print "oo"
375  print "pp"
376  print "qq"
377  print "rr"
378  print "ss"
379  print "tt"
380  print "uu"
381  print "vv"
382  print "ww"
383  print "xx"
384  print "yy"
385  print "zz"
386  print "aa"
387  print "bb"
388  print "cc"
389  print "dd"
390  print "ee"
391  print "ff"
392  print "gg"
393  print "hh"
394  print "ii"
395  print "jj"
396  print "kk"
397  print "ll"
398  print "mm"
399  print "nn"
400  print "oo"
401  print "pp"
402  print "qq"
403  print "rr"
404  print "ss"
405  print "tt"
406  print "uu"
407  print "vv"
408  print "ww"
409  print "xx"
410  print "yy"
411  print "zz"
412  print "aa"
413  print "bb"
414  print "cc"
415  print "dd"
416  print "ee"
417  print "ff"
418  print "gg"
419  print "hh"
420  print "ii"
421  print "jj"
422  print "kk"
423  print "ll"
424  print "mm"
425  print "nn"
426  print "oo"
427  print "pp"
428  print "qq"
429  print "rr"
430  print "ss"
431  print "tt"
432  print "uu"
433  print "vv"
434  print "ww"
435  print "xx"
436  print "yy"
437  print "zz"
438  print "aa"
439  print "bb"
440  print "cc"
441  print "dd"
442  print "ee"
443  print "ff"
444  print "gg"
445  print "hh"
446  print "ii"
447  print "jj"
448  print "kk"
449  print "ll"
450  print "mm"
451  print "nn"
452  print "oo"
453  print "pp"
454  print "qq"
455  print "rr"
456  print "ss"
457  print "tt"
458  print "uu"
459  print "vv"
460  print "ww"
461  print "xx"
462  print "yy"
463  print "zz"
464  print "aa"
465  print "bb"
466  print "cc"
467  print "dd"
468  print "ee"
469  print "ff"
470  print "gg"
471  print "hh"
472  print "ii"
473  print "jj"
474  print "kk"
475  print "ll"
476  print "mm"
477  print "nn"
478  print "oo"
479  print "pp"
480  print "qq"
481  print "rr"
482  print "ss"
483  print "tt"
484  print "uu"
485  print "vv"
486  print "ww"
487  print "xx"
488  print "yy"
489  print "zz"
490  print "aa"
491  print "bb"
492  print "cc"
493  print "dd"
494  print "ee"
495  print "ff"
496  print "gg"
497  print "hh"
498  print "ii"
499  print "jj"
500  print "kk"
501  print "ll"
502  print "mm"
503  print "nn"
504  print "oo"
505  print "pp"
506  print "qq"
507  print "rr"
508  print "ss"
509  print "tt"
510  print "uu"
511  print "vv"
512  print "ww"
513  print "xx"
514  print "yy"
515  print "zz"
516  print "aa"
517  print "bb"
518  print "cc"
519  print "dd"
520  print "ee"
521  print "ff"
522  print "gg"
523  print "hh"
524  print "ii"
525  print "jj"
526  print "kk"
527  print "ll"
528  print "mm"
529  print "nn"
530  print "oo"
531  print "pp"
532  print "qq"
533  print "rr"
534  print "ss"
535  print "tt"
536  print "uu"
537  print "vv"
538  print "ww"
539  print "xx"
540  print "yy"
541  print "zz"
542  print "aa"
543  print "bb"
544  print "cc"
545  print "dd"
546  print "ee"
547  print "ff"
548  print "gg"
549  print "hh"
550  print "ii"
551  print "jj"
552  print "kk"
553  print "ll"
554  print "mm"
555  print "nn"
556  print "oo"
557  print "pp"
558  print "qq"
559  print "rr"
560  print "ss"
561  print "tt"
562  print "uu"
563  print "vv"
564  print "ww"
565  print "xx"
566  print "yy"
567  print "zz"
568  print "aa"
569  print "bb"
570  print "cc"
571  print "dd"
572  print "ee"
573  print "ff"
574  print "gg"
575  print "hh"
576  print "ii"
577  print "jj"
578  print "kk"
579  print "ll"
580  print "mm"
581  print "nn"
582  print "oo"
583  print "pp"
584  print "qq"
585  print "rr"
586  print "ss"
587  print "tt"
588  print "uu"
589  print "vv"
590  print "ww"
591  print "xx"
592  print "yy"
593  print "zz"
594  print "aa"
595  print "bb"
596  print "cc"
597  print "dd"
598  print "ee"
599  print "ff"
600  print "gg"
601  print "hh"
602  print "ii"
603  print "jj"
604  print "kk"
605  print "ll"
606  print "mm"
607  print "nn"
608  print "oo"
609  print "pp"
610  print "qq"
611  print "rr"
612  print "ss"
613  print "tt"
614  print "uu"
615  print "vv"
616  print "ww"
617  print "xx"
618  print "yy"
619  print "zz"
620  print "aa"
621  print "bb"
622  print "cc"
623  print "dd"
624  print "ee"
625  print "ff"
626  print "gg"
627  print "hh"
628  print "ii"
629  print "jj"
630  print "kk"
631  print "ll"
632  print "mm"
633  print "nn"
634  print "oo"
635  print "pp"
636  print "qq"
637  print "rr"
638  print "ss"
639  print "tt"
640  print "uu"
641  print "vv"
642  print "ww"
643  print "xx"
644  print "yy"
645  print "zz"
646  print "aa"
647  print "bb"
648  print "cc"
649  print "dd"
650  print "ee"
651  print "ff"
652  print "gg"
653  print "hh"
654  print "ii"
655  print "jj"
656  print "kk"
657  print "ll"
658  print "mm"
659  print "nn"
660  print "oo"
661  print "pp"
662  print "qq"
663  print "rr"
664  print "ss"
665  print "tt"
666  print "uu"
667  print "vv"
668  print "ww"
669  print "xx"
670  print "yy"
671  print "zz"
672  print "aa"
673  print "bb"
674  print "cc"
675  print "dd"
676  print "ee"
677  print "ff"
678  print "gg"
679  print "hh"
680  print "ii"
681  print "jj"
682  print "kk"
683  print "ll"
684  print "mm"
685  print "nn"
686  print "oo"
687  print "pp"
688  print "qq"
689  print "rr"
690  print "ss"
691  print "tt"
692  print "uu"
693  print "vv"
694  print "ww"
695  print "xx"
696  print "yy"
697  print "zz"
698  print "aa"
699  print "bb"
700  print "cc"
701  print "dd"
702  print "ee"
703  print "ff"
704  print "gg"
705  print "hh"
706  print "ii"
707  print "jj"
708  print "kk"
709  print "ll"
710  print "mm"
711  print "nn"
712  print "oo"
713  print "pp"
714  print "qq"
715  print "rr"
716  print "ss"
717  print "tt"
718  print "uu"
719  print "vv"
720  print "ww"
721  print "xx"
722  print "yy"
723  print "zz"
724  print "aa"
725  print "bb"
726  print "cc"
727  print "dd"
728  print "ee"
729  print "ff"
730  print "gg"
731  print "hh"
732  print "ii"
733  print "jj"
734  print "kk"
735  print "ll"
736  print "mm"
737  print "nn"
738  print "oo"
739  print "pp"
740  print "qq"
741  print "rr"
742  print "ss"
743  print "tt"
744  print "uu"
745  print "vv"
746  print "ww"
747  print "xx"
748  print "yy"
749  print "zz"
750  print "aa"
751  print "bb"
752  print "cc"
753  print "dd"
754  print "ee"
755  print "ff"
756  print "gg"
757  print "hh"
758  print "ii"
759  print "jj"
760  print "kk"
761  print "ll"
762  print "mm"
763  print "nn"
764  print "oo"
765  print "pp"
766  print "qq"
767  print "rr"
768  print "ss"
769  print "tt"
770  print "uu"
771  print "vv"
772  print "ww"
773  print "xx"
774  print "yy"
775  print "zz"
776  print "aa"
777  print "bb"
778  print "cc"
779  print "dd"
780  print "ee"
781  print "ff"
782  print "gg"
783  print "hh"
784  print "ii"
785  print "jj"
786  print "kk"
787  print "ll"
788  print "mm"
789  print "nn"
790  print "oo"
791  print "pp"
792  print "qq"
793  print "rr"
794  print "ss"
795  print "tt"
796  print "uu"
797  print "vv"
798  print "ww"
799  print "xx"
800  print "yy"
801  print "zz"
802  print "aa"
803  print "bb"
804  print "cc"
805  print "dd"
806  print "ee"
807  print "ff"
808  print "gg"
809  print "hh"
810  print "ii"
811  print "jj"
812  print "kk"
813  print "ll"
814  print "mm"
815  print "nn"
816  print "oo"
817  print "pp"
818  print "qq"
819  print "rr"
820  print "ss"
821  print "tt"
822  print "uu"
823  print "vv"
824  print "ww"
825  print "xx"
826  print "yy"
827  print "zz"
828  print "aa"
829  print "bb"
830  print "cc"
831  print "dd"
832  print "ee"
833  print "ff"
834  print "gg"
835  print "hh"
836  print "ii"
837  print "jj"
838  print "kk"
839  print "ll"
840  print "mm"
841  print "nn"
842  print "oo"
843  print "pp"
844  print "qq"
845  print "rr"
846  print "ss"
847  print "tt"
848  print "uu"
849  print "vv"
850  print "ww"
851  print "xx"
852  print "yy"
853  print "zz"
854  print "aa"
855  print "bb"
856  print "cc"
857  print "dd"
858  print "ee"
859  print "ff"
860  print "gg"
861  print "hh"
862  print "ii"
863  print "jj"
864  print "kk"
865  print "ll"
866  print "mm"
867  print "nn"
868  print "oo"
869  print "pp"
870  print "qq"
871  print "rr"
872  print "ss"
873  print "tt"
874  print "uu"
875  print "vv"
876  print "ww"
877  print "xx"
878  print "yy"
879  print "zz"
880  print "aa"
881  print "bb"
882  print "cc"
883  print "dd"
884  print "ee"
885  print "ff"
886  print "gg"
887  print "hh"
888  print "ii"
889  print "jj"
890  print "kk"
891  print "ll"
892  print "mm"
893  print "nn"
894  print "oo"
895  print "pp"
896  print "qq"
897  print "rr"
898  print "ss"
899  print "tt"
900  print "uu"
901  print "vv"
902  print "ww"
903  print "xx"
904  print "yy"
905  print "zz"
906  print "aa"
907  print "bb"
908  print "cc"
909  print "dd"
910  print "ee"
911  print "ff"
912  print "gg"
913  print "hh"
914  print "ii"
915  print "jj"
916  print "kk"
917  print "ll"
918  print "mm"
919  print "nn"
920  print "oo"
921  print "pp"
922  print "qq"
923  print "rr"
924  print "ss"
925  print "tt"
926  print "uu"
927  print "vv"
928  print "ww"
929  print "xx"
930  print "yy"
931  print "zz"
932  print "aa"
933  print "bb"
934  print "cc"
935  print "dd"
936  print "ee"
937  print "ff"
938  print "gg"
939  print "hh"
940  print "ii"
941  print "jj"
942  print "kk"
943  print "ll"
944  print "mm"
945  print "nn"
946  print "oo"
947  print "pp"
948  print "qq"
949  print "rr"
950  print "ss"
951  print "tt"
952  print "uu"
953  print "vv"
954  print "ww"
955  print "xx"
956  print "yy"
957  print "zz"
958  print "aa"
959  print "bb"
960  print "cc"
961  print "dd"
962  print "ee"
963  print "ff"
964  print "gg"
965  print "hh"
966  print "ii"
967  print "jj"
968  print "kk"
969  print "ll"
970  print "mm"
971  print "nn"
972  print "oo"
973  print "pp"
974  print "qq"
975  print "rr"
976  print "ss"
977  print "tt"
978  print "uu"
979  print "vv"
980  print "ww"
981  print "xx"
982  print "yy"
983  print "zz"
984  print "aa"
985  print "bb"
986  print "cc"
987  print "dd"
988  print "ee"
989  print "ff"
990  print "gg"
991  print "hh"
992  print "ii"
993  print "jj"
994  print "kk"
995  print "ll"
996  print "mm"
997  print "nn"
998  print "oo"
999  print "pp"
1000 print "qq"

```

ARCHIVE procedure to produce this layout.



ARCHIVE procedure to complete the 'form'.

```

100  print "aa"
101  print "bb"
102  print "cc"
103  print "dd"
104  print "ee"
105  print "ff"
106  print "gg"
107  print "hh"
108  print "ii"
109  print "jj"
110  print "kk"
111  print "ll"
112  print "mm"
113  print "nn"
114  print "oo"
115  print "pp"
116  print "qq"
117  print "rr"
118  print "ss"
119  print "tt"
120  print "uu"
121  print "vv"
122  print "ww"
123  print "xx"
124  print "yy"
125  print "zz"
126  print "aa"
127  print "bb"
128  print "cc"
129  print "dd"
130  print "ee"
131  print "ff"
132  print "gg"
133  print "hh"
134  print "ii"
135  print "jj"
136  print "kk"
137  print "ll"
138  print "mm"
139  print "nn"
140  print "oo"
141  print "pp"
142  print "qq"
143  print "rr"
144  print "ss"
145  print "tt"
146  print "uu"
147  print "vv"
148  print "ww"
149  print "xx"
150  print "yy"
151  print "zz"
152  print "aa"
153  print "bb"
154  print "cc"
155  print "dd"
156  print "ee"
157  print "ff"
158  print "gg"
159  print "hh"
160  print "ii"
161  print "jj"
162  print "kk"
163  print "ll"
164  print "mm"
165  print "nn"
166  print "oo"
167  print "pp"
168  print "qq"
169  print "rr"
170  print "ss"
171  print "tt"
172  print "uu"
173  print "vv"
174  print "ww"
175  print "xx"
176  print "yy"
177  print "zz"
178  print "aa"
179  print "bb"
180  print "cc"
181  print "dd"
182  print "ee"
183  print "ff"
184  print "gg"
185  print "hh"
186  print "ii"
187  print "jj"
188  print "kk"
189  print "ll"
190  print "mm"
191  print "nn"
192  print "oo"
193  print "pp"
194  print "qq"
195  print "rr"
196  print "ss"
197  print "tt"
198  print "uu"
199  print "vv"
200  print "ww"
201  print "xx"
202  print "yy"
203  print "zz"
204  print "aa"
205  print "bb"
206  print "cc"
207  print "dd"
208  print "ee"
209  print "ff"
210  print "gg"
211  print "hh"
212  print "ii"
213  print "jj"
214  print "kk"
215  print "ll"
216  print "mm"
217  print "nn"
218  print "oo"
219  print "pp"
220  print "qq"
221  print "rr"
222  print "ss"
223  print "tt"
224  print "uu"
225  print "vv"
226  print "ww"
227  print "xx"
228  print "yy"
229  print "zz"
230  print "aa"
231  print "bb"
232  print "cc"
233  print "dd"
234  print "ee"
235  print "ff"
236  print "gg"
237  print "hh"
238  print "ii"
239  print "jj"
240  print "kk"
241  print "ll"
242  print "mm"
243  print "nn"
244  print "oo"
245  print "pp"
246  print "qq"
247  print "rr"
248  print "ss"
249  print "tt"
250  print "uu"
251  print "vv"
252  print "ww"
253  print "xx"
254  print "yy"
255  print "zz"
256  print "aa"
257  print "bb"
258  print "cc"
259  print "dd"
260  print "ee"
261  print "ff"
262  print "gg"
263  print "hh"
264  print "ii"
265  print "jj"
266  print "kk"
267  print "ll"
268  print "mm"
269  print "nn"
270  print "oo"
271  print "pp"
272  print "qq"
273  print "rr"
274  print "ss"
275  print "tt"
276  print "uu"
277  print "vv"
278  print "ww"
279  print "xx"
280  print "yy"
281  print "zz"
282  print "aa"
283  print "bb"
284  print "cc"
285  print "dd"
286  print "ee"
287  print "ff"
288  print "gg"
289  print "hh"
290  print "ii"
291  print "jj"
292  print "kk"
293  print "ll"
294  print "mm"
295  print "nn"
296  print "oo"
297  print "pp"
298  print "qq"
299  print "rr"
300  print "ss"
301  print "tt"
302  print "uu"
303  print "vv"
304  print "ww"
305  print "xx"
306  print "yy"
307  print "zz"
308  print "aa"
309  print "bb"
310  print "cc"
311  print "dd"
312  print "ee"
313  print "ff"
314  print "gg"
315  print "hh"
316  print "ii"
317  print "jj"
318  print "kk"
319  print "ll"
320  print "mm"
321  print "nn"
322  print "oo"
323  print "pp"
324  print "qq"
325  print "rr"
326  print "ss"
327  print "tt"
328  print "uu"
329  print "vv"
330  print "ww"
331  print "xx"
332  print "yy"
333  print "zz"
334  print "aa"
335  print "bb"
336  print "cc"
337  print "dd"
338  print "ee"
339  print "ff"
340  print "gg"
341  print "hh"
342  print "ii"
343  print "jj"
344  print "kk"
345  print "ll"
346  print "mm"
347  print "nn"
348  print "oo"
349  print "pp"
350  print "qq"
351  print "rr"
352  print "ss"
353  print "tt"
354  print "uu"
355  print "vv"
356  print "ww"
357  print "xx"
358  print "
```


steps to the next clean sheet. Archive acts in a different way from Basic as characters below 31 are interpreted as special screen driver codes which may be found by trial and error. Many of the codes produce an error message "i/o failed" which has both puzzled and deterred many users from taking advantage of the Archive in-built screen handling. The only authoritative book I have found is the manual for *Xchange* running on an IBM PC or clone. None of the QL Archive books covers the screen drivers or the later versions of Archive.

Screen matrix

The Archive screen in mode 0,8 consists of a matrix 80 wide by 24 down. To address this matrix it is possible to use the screen driver of Archive and utilise these special effects to the full. If you tried it and have obtained what appears to be an error by sending chr(9), the only error is that you told Archive to do something without supplying the parameters for it to function correctly.

Great stress is laid on this aspect of programming when SuperBasic FUNCTIONS are discussed, with it being emphasised that FUNCTIONS usually need parameters. Archive character codes are no different in that those below 31 need, in most instances, one or more parameters, as in general terms they mimic some SuperBasic functions. CLS is the same in SuperBasic as CLS in Archive but it appears that WINDOW, width, depth, start point down, start point across is absent from Archive. Notice that "WINDOW" in SuperBasic must have all four parameters supplied for it to function without returning an error message, exactly as chr(20) in Archive must have the four parameters supplied. In this windowing example the equivalent are:

SuperBasic	Archive
10 WINDOW 1,512,	PRINT chr(20)+chr
256,0,0,	(0)+chr(0)+chr(80)
	+chr(22)

It looks cumbersome but it needs typing in only once in a procedure such as:

```
proc windo1
  print chr(20)+chr(10)+chr(2)+chr(60)
  +chr(18)
  print chr(2)+chr(4): rem Sets paper to
  green
  cls
```

Notice that chr(20) has to be sent first and all the four parameters relating to the four top left start points as well as width and depth for it to be successful. As further examples, chr(5) toggles underline on and off, therefore no parameters are needed, while chr(2) sets the paper colour provided the single paper colour parameter is supplied.

If a high-numbered — above 224 — character is sent to the screen in Archive it may draw the corner of a box, as there are box drawing characters at those locations. The same character spooled to the

Machine code interface

Machine code modules can be incorporated into ARCHRTM to speed slow parts of the coding by means of the additional usr() function and a use qualifier to the LOAD command. The code must be re-locatable and in addition the file must contain a six-byte header with the following format. Bytes 0 to 3 must contain "pmc0" with bytes 4 and 5 being a word giving the length of the following code.

Example

	dc.b	'pmc0'
	dc.w	last-first
first		
	lea	sound(pc), a3
	move.b	d0, note-sound (a3)
	moveq	£\$11,a0
	trap£1	
	rts	
sound		
	dc.b	\$a,8
	dc.1	%1010101010101010
note		
	dc.b	160,0
	dc.w	0, 1000
	dc.b	0,0,3
last		
	end	

Archive functions

count	eof	memory	errnum	recnum	numfld	fieldv	fieldt
fieldn	value	num	dec	gen	found	upper	lower
inkey	getkey	usr	let	rem	all	endall	if
while	endwhile	else	endif	import	export	print	lprint
spoolon	spooloff	input	mode	cls	proc	endproc	return
local	error	stop	new	quit	use	open	look
create	close	update	first	last	next	back	position
locate	find	search	continue	order	alter	select	reset
append	delete	display	insert	lilst	edit	save	load
merge	run	sedit	sload	ssave	screen	sprint	sinput
trace	dir	format	kill	dump	backup	endcreate	as
logical	quill	tab	at	ink	paper	object	protect
usr							

printer by Archive will not draw the box corner and may do nothing or something which appears to be stupid, such as printing in italic. It is not stupid, as the printer is just doing what you told it to do.

On an Epson printer the same box drawing facilities are available as those produced on-screen by the high number characters in Archive but the printer needs several instructions for them to be interpreted correctly. First it must be set to graphics mode with ESC, 'm',6; then the correct code, e.g., 136, sent to make it print a box corner.

Printer driver

There is no problem if the printer driver translates are installed correctly in printer_—dat or some other method of driving the printer is used. In my opinion it is worth the effort, as it gives a professional-looking form on the screen and as on paper.

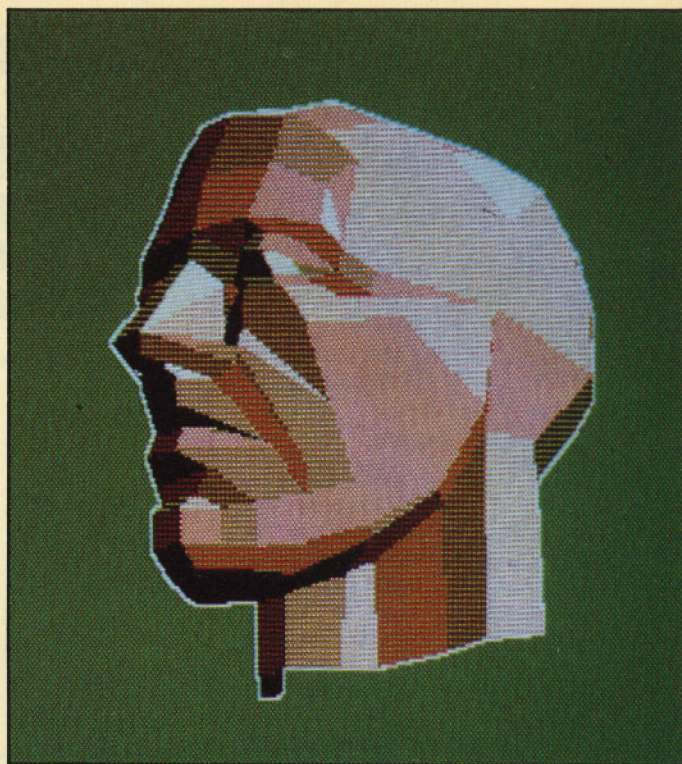
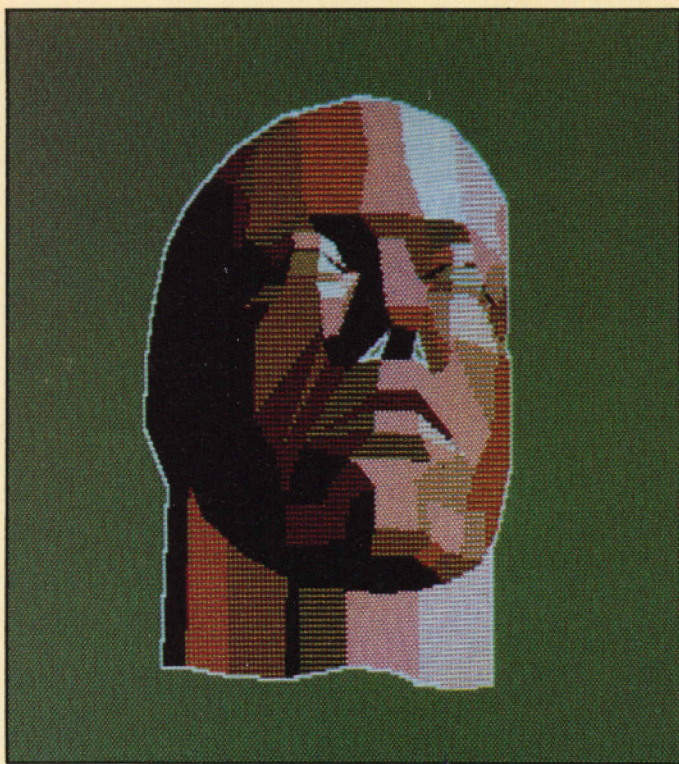
The difference between Archive _prg files and _pro files is that the _pro files are TOKENISED, as against the _prg files being virtual ASCII. Neither type of

file is machine code, as they are both interpreted at run time but there is a facility for incorporating machine code modules to speed some of the slower parts.

RTM

This brings us to the Run Time module, RTM, which is supposed to need Archdev to program it. It does not. Later versions of Archive are successful, as is Archive running under Xchange. Much of this is unnecessary if a text editor is used and even Quill can be used to type-in the text then EXPORT to it Archive. Remember it is only a straightforward text file, as is a SuperBasic program file. With this premise it then follows that recovery of a still-open Archive _dbf file is a job for a reasonable text editor. Chas Dillon's *Editor* is excellent for tackling this type of problem and both it in all its versions and Spy make the job easy.

A brief run-down of the Archive control codes follows but as the whole explanation occupies some 12 very full pages it is not possible to reproduce every detail.



The Animated Head by Mark Swift of Blackley, Manchester impressed everyone who saw it. "A polished entry of classical feel and quality," said the judges. As the head rotates smoothly, the shading changes to reflect the light source,

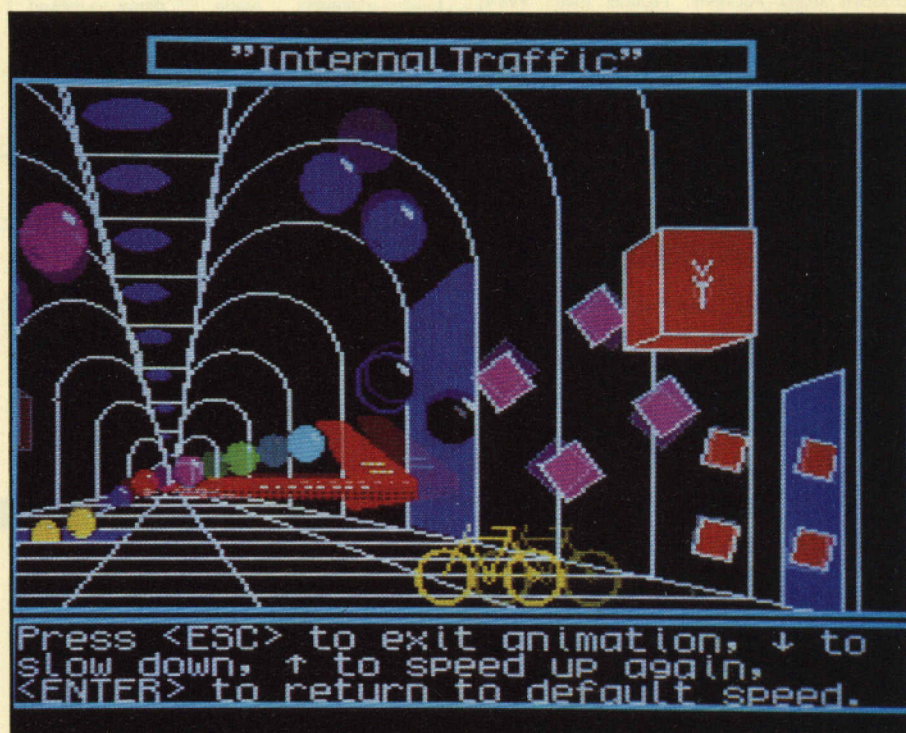
achieving a remarkable solidity of image. If there is a graphic which shows off the potential of the QL, this is it." The program needs 512K of extra memory to run. So the QL can still turn heads...

QL ARTIST OF THE YEAR 1988

At last we can reveal the results – and the varied and beautiful output – of the *QL World Artist of the Year 1988* Competition.

After much deliberation Magnetic Media finalists made several postal journeys around the country – the judges decided that the graphic display which best represents the QL is *The Animated Head* by Mark Swift. He wins a Phillips CM8833 colour stereo monitor, supplied by Sector Software.

The runner-up, Mark Knight, will be offered a choice of Sector software. The standard of entry was very high, and we look forward to running another art competition in the future.



Internal Traffic by Mark Knight of London was praised for its effective use of moving figures, humour and careful programming. "This man has put a great

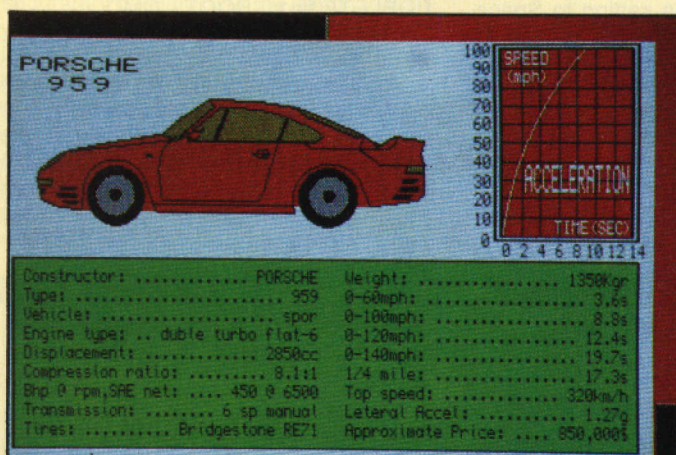
deal of work into making the whole scene, and impresses with his skills," say the judges. The program uses a DIY Toolkit routine, Turbo and needs 640K.



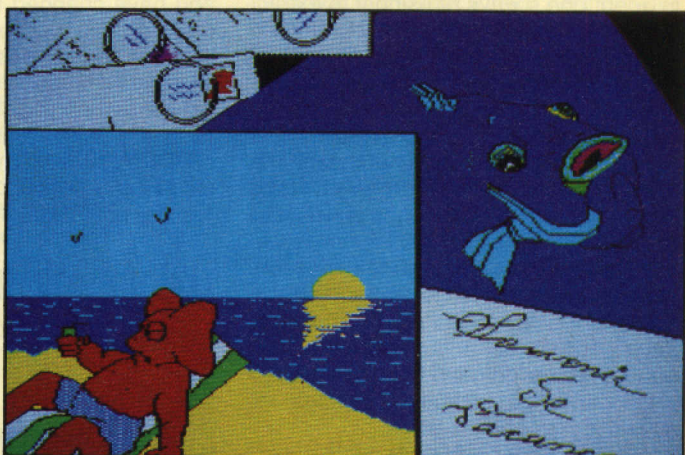
Plant Life by C. Bate from St. Helens, Lancashire builds ingenious trailing plants using fractals. The "flower arrangement" that grows is a little different each time the program runs.



"The maths he used must have been a headache, and the program, though slow, is fascinating to watch," said the judge. It is also pretty.



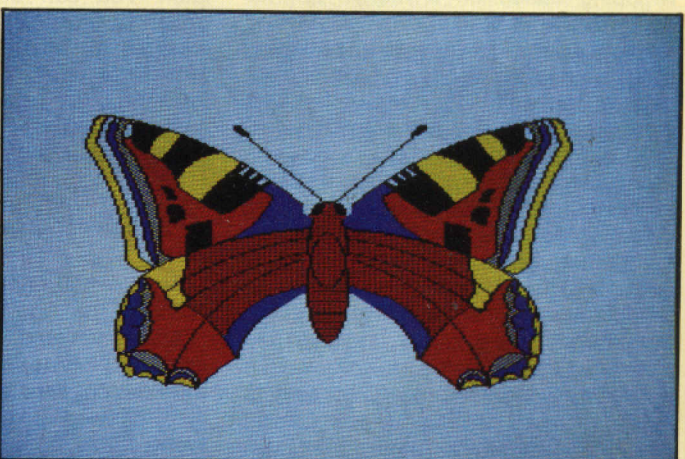
Porche 939 by Ampridikis Hardlampos of Drama, Greece has been copied painstakingly from a magazine photograph using a hand-drawn transparent overlay and a black and white monitor. The result is a tribute to sharp eyes and patience.



Tacot/Music/Holidays by Saby Chrystiane of Clermont Ferrand, France was commended for blending careful artwork with a computer environment. It uses *Eye-Q* and *GraphicQL*, with a simple SuperBasic 'flipper' to present one screen after the other.



Sinclair QL by S. Usher of Helston, Cornwall, used a fast-moving series of road signs, Mandelbrot patterns and slogans to praise the QL; one or two slogans were felt to be a little risky but the presentation is provocative and effective.



Butterfly by P. Harris of Llangynog, Carmarthen, is an attractive static picture designed in low-resolution mode with *Eye-Q*, with a very thoughtful use of a few colours, closely inspired by, but not copied slavishly from, nature.

YOU CANNOT

These comments are orientated towards use of *The Editor* (Special Edition) for word-processing. The program did not start out designed for this job but development in the course of a year or so made it into more than just an alternative to Quill. It was much faster, had a wider range of commands, and it could be used, when the occasion demanded, for manipulating program files, something Quill was unfitted to do.

As it has probably reached the end of its development cycle, it makes more sense to adapt one's ways of using it to extract the best effects, rather than to bombard Digital Precision with suggestions for improvements – much as I would still like to see certain changes made.

EDT_CONFIG.BIN. This routine sets the basic operating parameters for the program. As you cannot run it from within Editor, the settings have to be considered as relatively permanent, so they need to be suitable for most of your usage. Temporary changes, however, can be made to many parameters by using the **BOOT_CMD** file and/or the **RC** command; there is no need to feel obliged to stick with the configured settings for everything you do.

Right margin

An example of a desirable permanent setting is a right margin of 256, the maximum. This allows handling of programming files without undesirable word-wrapping based on a margin set for document work. It is easy to go beyond the right edge of the screen when writing lines of SuperBasic code and they will not perform too well if they are saved with numbered lines separated by the tail-ends of other lines.

You can set the right margin at whatever is suitable for your text processing – 83 utilises the full screen width – and then reset it during operation whenever necessary, using the **"SR xxx"** command. If you produce spreadsheet-like files, for printing in condensed typeface you might find the default setting for Horizontal Pan Percentage unsuitable; try 55 percent as a start-point. When Editor shares the QL with other programs, the cursor can become sluggish and setting Cursor Speed to 1 is desirable; if extra speed is needed only occasionally, set it via the **BOOT_CMD** or other command file. On the subject of that file, you can change the file name for which Editor looks when

Every rule book has another set of rules which are not so easy to find. This month Bryan Davies looks at word processing commands for *The Editor* (Special Edition) and gets the most from them.

starting, using the Name of Boot Command File option.

It may be just an oddity of mine but I dislike the default characters used as Word Delimiters for Cursor Movement and have changed the setting to Space only; as you cannot get the routine to accept a space on its own you have to put in another character in addition, preferably one which is unlikely to occur in a text file – e.g., Shift+Enter.

BOOT_CMD. Every time Editor is loaded, this file is looked for. If not found, loading continues. If it is found, whatever

"It makes more sense to adapt one's ways of using it to extract the best, rather than bombarding DP with suggested improvements."

commands are contained in it are carried-out. This is your chance to save work later. You can use the configuration program for relatively permanent changes to the set-up but the boot provides a more convenient way of making temporary, quick changes, and of setting parameters which cannot be set during configuration.

Colour combinations

If you have a colour display and have reached the age when eyesight is a constant irritation, why not experiment with the colours used by Editor, to achieve the best combination for your eyes and viewing conditions? I keep reverting to green characters on a black background. If you use two copies of Editor during the same session, why not identify them by different colour schemes? There is no reason to limit yourself to one **BOOT_CMD** file; like **PRINTER_DAT** files with Quill, you can write a small

routine to select the required **BOOT_CMD** file before starting Editor. Put lines like this into your system boot file:

```
1 DELETE flp1_BOOT_CMD
2 PRINT, "Do you want BOOT_CMD 1
  or 2?"
3 LET a=INKEY$(-1)
4 SELECT ON a
5 I=1: COPY flp1_BOOT_CMD1 TO
  flp1_BOOT_CMD
6 I=2: COPY flp1_BOOT_CMDS TO
  flp1_BOOT_CMD
7 END SELECT
8 EXEC flp1_Edt_bin
```

or use **"COPY_O flp1_BOOT_CMD, flp1_BOOT_CMD1"** if you have a Toolkit.

and make **BOOT_CMD** set the parameters you want. For example:

M120 KM 4,0 KC 7,0 KE 2,7 CD1

will set the memory space for a document to 120KB, make the text colour green against a black background, the command-line text white against a black background, and the error text red against a white background, and then set the cursor speed to maximum.

Where documents vary considerably in size it is worth considering using a command like **"M120 M20"** to set the initial memory space allocation high, then drop it to a level sufficient for average documents; if additional space is required later it is more likely you will get it if space has been reserved initially than if you had started Editor with the default 15KB allocation. Finding additional memory can be a problem when you run other programs at the same time; there may be plenty of memory there but not in a block usable by Editor.

While windows may look striking, there is little practical point in using less than the full screen space for text input, so why not take advantage of the QL "extra width" and set the Editor window for the maximum 83 characters? Losing the odd line

MEAN

vertically because of indicator windows, such as the clock and caps/memory indicators shown in the figure, is not a great handicap and causes less waste of time than having to scroll horizontally to get to the ends of lines.

Two copies

When two copies are in use, and copying is done between them, they can be set full-width one above the other. Your main copy can be left full-screen and the secondary one laid over the bottom of it. If you have a Toolkit with the ALTKEY function, changing the size of Copy 2 can be done with two keypresses. Set the size of Copy 1 with the configuration program, so that copy loads with no need for modification, then determine how many operations of the ALT and cursor keys are needed to alter the size and position from that of Copy 1 to what is required for Copy

F4, then ALT+up cursor nine times – to reduce the window size – then down-cursor nine times – to move the window to the bottom of the screen. Assuming Copy 1 was full-screen, keying ALT+/ should result in Copy 2 being set up to be roughly half-screen high and based in the lower half of the display. Being able to see text in two files – or two parts of the same file – at

“There is merit in making a back-up copy to Microdrive when you use disc as the main medium; a problem with the drive or interface can put you out of business”.

the same time makes copying from one to the other easier; use RAM disc as temporary storage, mark the required block, and write it to RAM with the “BW.RAM1__temp” command, then transfer (CTRL+C) to the other copy of Editor, cursor to the place where the block

```
1989 May 19 11:04:06 Copy off / memory = 124928
SL1;S11;SR00;TR;T122;T126;T177;CD20;R.SPList
;PPL60
;PLM10
;PFT1
;ISSUED -89 Page ##0
;SPARE PARTS LIST0
;B0
;Many parts are common to these models; the ;MODELS column shows which models
;use each part. The model codes are given in brackets after the Service Numbers.
;Parts marked ;BNS are not shown on the parts breakdown drawings.
;Parts marked ;BNA are not available as replacements. Contact Service Department
;about compressors and parts with -- for the Price Group.
;
;ORDER NUMBER: PG: DESCRIPTION: MODELS:
;
;Cads: RC.SPL
```

2, and set up an appropriate ALTKEY to perform those keypresses after Copy 2 has loaded. In the initial boot program, put a line like:

```
ALTKEY '/,CHR$(246)&FILL$(CHR$(209),9)&FILL$(CHR$(216),9)
&CHR$(10)
```

This is equivalent to pressing SHIFT+

is required and insert it with the “AF.RAM1__temp” command.

COMMAND FILES. No doubt many users never utilise the RC command but it is one which can save much time. The BOOT_CMD file is a normal command file, except insofar as the program calls it automatically when it starts up. The program stores certain configuration information but that does not include the

format – layout – of individual documents. A command file can store the format details and re-load them immediately prior to the loading of the document. Under some circumstances, the flexibility given to file structure by not keeping information such as margin settings in files is very useful but it tends to be an inconvenience if all your work is word processing.

Figure one is a screen dump, showing a command file (top line) and the document file it loads. The document – “SPList”, line 3 onwards – needs three tabs and the right margin setting; the TR is to ensure that any other tabs set prior to loading this document are removed.

Likewise with the SL and SI commands. The CD command can be dispensed with if the value set during initial configuration suits all jobs and system set-ups the user has but it can be useful sometimes. Line 1 will appear briefly on the status line when the command which is shown there on the screen dump starts executing; it is shown at the top of the text screen here only for illustration and neither it nor line 2 are part of the master document, which starts at line 3.

Lines in a document file beginning with a semi-colon are treated by the printer program as being comments only – similar to REMark in SuperBasic – unless the letter-group following the semi-colon happens to be that for a print command. A few such lines at the top of a document can be good memory-jerkers. Normally I put the file name, number of words, then size, on the first line and the identifying numbers of the disc and cartridge on which copies are kept on the second line.

Back-ups

There is merit in making a back-up copy to Microdrive when you use disc as the main medium; it takes only a problem with disc drive or interface for you to be out of business but the Microdrives can normally be expected to continue functioning in such circumstances. If any comments need to be made relative to the document, its format and printing, they can come next.

On the screen dump, the first two lines of the document proper are partly reminders of what the page length and left margin settings are, but the printer program, unlike The Editor, acts on those lines, so they can be used to over-ride the default settings in the DRIVER_DAT file at print time.

Bear this in mind when making comments, because it is possible for the print program to take them as commands if the first few characters match one on its list. Remember that a command put into a file for the benefit of the printer program may also need to be executed when the document is being edited, to avoid possible confusion.

When the basic page length setting in the DRIVER_DAT file is PPL60 and you

choose to put the command line ";PPL65" into the document, the document will not be re-formatted to 65 lines per page unless you use either the command F3 PPL65 ENTER or put the cursor on that command line and use F3 EX ENTER. By using the latter command you can make a command sequence part of the document; put the commands after a semi-colon, in the manner you would use on the normal command line – when using F3 – then load the file and position the cursor somewhere on that line and key F3 EX ENTER. For example:

```
;QLW99(2344/21650)
;D69/C119
;pages not consecutive – don't print from
12A
;needs spell-checking
;PLM10 SL1 SI1 SR80 PPL60 PFM4
;PFT1 DATED PAGE ##
QL WORLD
```

The fifth line is the one on which to use EX.

LONG DOCUMENTS. While the Document mode enhances the word processing capability of the program considerably, it has some unwelcome side-effects. The most obvious is a considerable slowing of operations. This is particularly noticeable when using the program in a multi-tasking environment; operations such as writing typed characters to the screen can become so slow that you may have to wait several seconds for it to catch up with your input, even if you are not a fast typist; you can improve the situation considerably by installing the Lightning "accelerator".

Long documents

Long documents as used here mean about 1,000 words – 5KB file size – upwards on set-ups which use all of 896KB of memory to hold several major programs at the same time.

The message is not to use the MD or RD commands unless they are essential; do as much of the input and alteration work as possible before thinking about splitting the document into pages. Even when the document is in its final state, consider making changes, however minor, without using RD. The time saved can be considerable. If you make use of the SQ command lock-ups can occur when in Document mode, so use it in Text mode; additionally, the Block commands can produce error messages and a return to the top of the document, which is infuriating if occurring frequently when you are well into a multi-page document.

It may seem a backward step but it is best to save the document, then read it in with the R command, before marking a block and ordering it with the SQ command.

REFORMATTING. The program was not designed to re-format text as you type

it in and this is one reason operations are so much faster than with Quill. The PR command is distinctly slow; use it sparingly and make sure before using it that there are blank lines between blocks of text which you want to keep separate. You can find yourself bouncing off the ceiling if lack of attention to this point results in a document you have re-worked laboriously – perhaps from a format used by another program – becoming scrambled. Always save the document as the last act before using PR.

Watch for incorrect page lengths when PR is used; it can cause the page length to become one line more than the set PPL, if PR is used over a Page Break. This may become apparent only when you print and can cause wasted time sorting through a long document to find the problem area. You may have to force the "extra" line into the next page using Enter and then use the PR command again from the top of that next page. Using the PPL command may not do the job completely, because the line at the end of the page will not have been re-formatted correctly.

There are two ways offered for printing, one taking much less time than the other. You can use the "WP" command when in Editor and make a quick print without exiting the program. This is rather like using the Quill Print command without the program disc/cartridge being in drive 1; all text is printed in the basic printer font, usually 10-pitch Pica, without enhancements such as Bold and Underlined. If this is sufficient for your needs, use it, because it is much more convenient than the alternative. To get a left margin, go to SuperBasic and type-in – perhaps set an ALTKEY for – a line such as:

```
OPEN#7,SER1:PRINT#7,CHR$(27)&
CHR$(65)&CHR$(27)&CHR$(108)&
CHR$(10):CLOSE#7
```

This will re-set Epson-compatible printers to the power-on settings, then set the left margin to 10 characters – 1in. with 10-pitch. You can also set typeface and

"Watch for incorrect page lengths when PR is used; the page can become one line longer than the set PPL."

enhancements this way. Any special codes present in the document are likely to produce "permanent" changes in the printout, so try to avoid putting them in until draft printing has been completed. A call for condensed print in the address on a letter can result in the whole letter being printed condensed and, besides that, the printer will be left set for condensed when the next print request is made.

EDTPRT__BIN. This separate program allows a wide range of customisation of printed output. You have to step out of Editor to use it but you can use CTRL+C to switch between the programs. While the program code of later versions is less than with earlier ones, the memory space required is still almost 50KB and you need to remember that when setting the allocation for Editor. An increase in the latter to cope with a large document might leave you short of space for the print program. There are two places where you can put print parameters – into the document or in the DRIVER__DAT file. The latter is for basic, semi-permanent settings, which can be altered as required in the document.

Not continuous

A typical case of where changes might be required is page numbering. Perhaps some pages of a document – drawings, for instance – are not prepared on the QL and the page sequence is not continuous. My experience has been that you need to experiment with the insertion of parameters into the document file, because the effects are not always what you would expect from reading the manual. In theory, the first page which should have a page number which is out-of-sequence needs the line ";PPN n" somewhere in that page, where "n" is the number you wish to see printed on it.

In long files, I have found it necessary to put "PPN n+1" on the page which has to be numbered "Page n"; this may be peculiar to certain files, lengths or page numbering arrangements, since putting "PPN n" at the top of a page in a small file correctly produces "Page n". Unfortunately, it may also produce "Page n-1" as the number for the previous page.

Try inserting the PPN command at about the middle of the page and make the number one more than that page needs to be numbered; this seems to produce the correct page numbering for that page and subsequent ones and avoids the prior one being re-numbered also.

You need to keep awake when selecting pages to be printed; ask for pages by their re-numbered designations. Another oddity is that the "Proceed within same file" option after a page or group of pages has been printed may offer as default the – numerically – next page for printing but it then ignores a request to print that and proceeds to offer the one after that, and so on.

The only way I have found to circumvent this is always to print sequences of pages and make a separate operation of printing any pages not in a sequence. There is another reason for this; selecting a page which is not the next one on offer can lead to a print being obtained with the incorrect page number on it; again, the only solution seems to be to ask for that particular page first, not from the "Proceed within same file" option.

BASIC

improvements

Most people to whom I speak who do their own programming do not seem to use functions, do not understand them and would not touch one with a bargepole. I do not understand why. They can be more useful than procedures and they can certainly produce some very elegant code.

I did not use them for a long time. I think it may be something to do with the fact that they return a value of some kind. It may be better to describe them as something which Has a value rather than Returning one.

A value

What is a function? In mathematics, a function is an equation which returns a value, so perhaps that confirms from where the name came. So far as programming is concerned it is a sequence of instructions which on being executed can be considered as having a value. You have probably seen the function which returns a square of a number. Just to be different, here is one which returns the square root:

```
100 DEF FN sqr__root(x)
110 IF x<0 THEN PRINT "Don't be silly":
RETURN 0: ELSE RETURN x ^ .5
120 END DEF sqr__root
```

Now, how do you use it? Like this:

```
PRINT sqr__root 99 or
result=sqr__root 99
```

For want of a better way of putting it a procedure is like a verb. It does something. A function is like a noun. It is something. Do not try to continue the analogy too far, because English is a much more sophisticated language than SuperBasic.

In the foregoing example, PRINT is the equivalent of a verb and sqr__root is like a noun – has a value or meaning. The value is the result of executing the code in the function, using any values given to the function as parameters.

A parameter passed to a function, or procedure for that matter, is treated as a local variable in the function. This means that the (x) used as a parameter to sqr__root has no meaning outside the

function unless it is defined specifically. Look at these PRINT statements:

```
PRINT 'a'           :This is lower-case a
PRINT a$            :This is a string variable
PRINT CHR$(a)       :a is a numeric variable
PRINT sqr__root(a)   :So is this
```

In the last two, you will notice the similarity. CHR\$ is a function and is used in exactly the same way as the sqr__root function. The main difference between CHR\$ and sqr__root is that CHR\$ is included in the QL/Thor, whereas you wrote the other one. I assume that you are satisfied using CODE, CHR\$ and so on, so why not write some of your own?

Look again at sqr__root. It does not only return a value. If you enter something silly it also prints a string. There is no

Most people whom I speak would not touch a function with a bargepole, yet functions can be more useful than procedures and certainly produce some very elegant code.

reason why a function cannot do this, although many lesser Basics allow a simple calculation only as a function.

In fact, in SuperBasic, a whole program could be a function, provided it returns a value of some kind before it finishes. Here is a useful function:

```
150 DEF FN range(item,lo,hi)
160 IF item>=lo AND item<=hi THEN
RETURN 1:Else RETURN 0
170 END DEF range
```

This was used as an example in the first article in the series. If you recall the part of that article which mentioned boolean

In the second instalment in his introduction to better Basic, Desmond Barry flies the flag for functions.

variables, you may recognise that 'range' is a boolean function. Remember that it can be used just like a variable. In this case, it can have only two values, either 1 or 0. To refresh your memory, it is used like this:

```
240 n=CODE(INKEY$(#15,-1)):IF
NOT range(n, 49,56) THEN GOTO 240
```

Are you getting the hang of it? Functions do not need to be numeric. I use one called trunc\$(a\$) which returns a\$ with trailing spaces removed. You use it by:

```
text$=trunc$(text$)
```

You will have noticed that the 'type' of the function is the same as the 'type' of value returned. In SuperBasic only three 'types' are supported – strings, floating point numbers and integers. Thus, trunc\$ returns a string as its value and range returns a floating point number. The only thing to consider between floats and integers is that some aspects of SuperBasic do not accept integers, notably FOR loops.

The functions I have mentioned so far have been fairly simple and straightforward. Now I will describe a rather more complex one. I call it select__file\$. It has one parameter, device\$. It is used by:

```
200 filenames$=select__file$(device$)
```

It does a directory of device\$, displays them in a useful format, then allows you to move round a large cursor like the Abacus one, highlighting filenames. When you find the one you want you press ENTER and that filename is returned as the value of the function and assigned to filename\$.

Not simple

The following routine took me a long time to write and it is a part of one of my commercial programs, so I regret that I cannot publish the code. I will give a description of how it works. You may care to try writing it yourself.

```
DEF FN select__file$(device$)
```

do a directory on device\$
get the directory into an appropriate

array
display it in a regular format on-screen
highlight the first filename
LOOP

cursor keys to move highlight
if ENTER then return currently
highlighted filename
END LOOP
END DEF select__file\$

Obviously, this is not a simple function. It is probably more like a procedure except for the RETURN statement. If it was a procedure you would probably set filename\$ inside or just outside the procedure, then use it to do whatever you needed to get it for. All very complicated. See how it is used to print a file:

```
340 DEF PROC print__file
350 INPUT "Device? ";dev$
360 REM error check here for validity
370 PRINT "Which file?"
380 file$=select__file$(dev$):REM this is
the function
390 COPY dev$&file$ TO serir
400 END DEF print__file
```

Of course, it can be used in the same way anywhere you need to get a filename. Then you write a function to return the device name as well and you do not need to type the things at all.

We have now covered two simple functions and described one very complex one. Now for something rather unortho-

dox. Suppose you want to do something like the select__file\$ problem of returning a filename. You have a list of items identified by a number and a string value, e.g.:

1 Lettuce
2 Tomato
etc.

For the purposes of this, you need to return the reference number as well as the string. There is nothing to prevent you using a string function which returns the number as well as the string. All you need to do is to use the QL/Thor coercion to turn 1 into '1'. In the case of the foregoing items, it would be RETURN 1&'Lettuce', 2&'Tomato', etc.

Split string

You will note that this is satisfactory for situations where you have up to nine items, because you can split the string outside the function. What if you have more than nine items? When I encountered this problem I concluded that, rather than pad and concatenate strings, I would add 10 to the reference number, then coerce it into the string. The process can be reversed outside the function.

In this case you are using a function to return two values, a number and a string. Show me in the manual where it says you cannot.

A variation on this is if you need to return two numbers. If you multiply one of them by a factor equal to one more than the maximum the other can be, they can be treated as a single number and separated outside the function again.

Suppose you want to return the area and perimeter of a square, which could be up to the size of the screen, i.e., 512 by 256 pixels:

The maximum area in pixels is $512 \times 256 = 131072$.

The maximum perimeter is $2 \times (512 + 256) = 1536$.

Your RETURN statement will be:

RETURN area*1536+perimeter

The function is assumed to assign its value to a variable called value. e.g., value=FName. When you get out of the function, you separate them by:

area=INT(value/1536):perimeter=value-area

I know this is a silly example because it would be easier not to use a function here but it illustrates the principle.

I hope some of this article has whetted your appetite for functions. They can be some of the most elegant structures in programming in any language which supports them. They can really improve your SuperBasic.

TF Services

London STD codes

London 01- dialling code is changing to 071- or 081- in May 1990. If you have a computer based address list, we can provide the data you need to change your lists, and even a program to change text files (eg Psion EXPORT files).

Disks contain 5 text lists in database import formats (quote delimited, etc) & a program for text (ie export) file conversion. Also provided:

- IBM PC : Files for major databases/spreadsheets
- ATARI ST : Psion Archive .dfl file
- ATARI ST : The 5 text lists & program only

3.5" 5.25 disk or microdrive - £10

COMPUTER CLEANER

Much improved mains filters, 40-80 db cut & 3 spike filters. 390 joule cut (3 & 4 way)

Certified X/Y 240ac capacitors

2-way (5a adaptor).....	£14
3-way (5a adaptor).....	£18
4-way (13a trailing socket).....	£24

QUICK QL REPAIRS

VAST stock of components & working circuit boards to ensure quick and reliable repair (Usually same day)

DIs tested with Thorn EM test rig and ROM software

(MOV hardware extra if required)

£25 (6m gntee)

Qualsoft Terminal

Viewdata/VIS2/ASCII QL TERMINAL EMULATOR

'Een deluxe communicatieprogramma van Qualsoft'

Multitasking program for electronic mail, PRESTEL etc. Phone directories for ALL (yes ALL) modems for the QL, autodial (where poss) and logon, two-way FILE TRANSFER to ATARI ST, IBM PC, Psion Organiser (via comes link), XMODEM, real time clock/timer, buffered logs to file/printer/transit files, editable command line, Swedish/Horwegian options, EOL translates, editor etc.

SOLVES ALL PACKAGED MODEM SOFTWARE PROBLEMS

Unbuffered modems usable with Miracle Modemport

Software (3.5" or ndw) & manual **£30**

FILE TRANSFER

Programs to transfer text AND programs error free (XMODEM) between computers

Available for IBM PC and compatibles, ATARI ST and Sinclair QL. Connects to Psion organiser via Psion comes link

Qualsoft program (per w/O)..... £7.50

Serial lead (max 2 computers)..... £10

Complete package for 2 computers, **£25**

ASTRACOM Modem

Intelligent buffered modem with text status messages. Hayes protocol, parallel printer port (can be used as 6k serial to parallel printer buffer), 240vac or 9-12vac, Autodial/answer and many programmable registers incl printer logging of RX and/or TX data, BT approved. Can be used with any computer.

U21/23 (300/300 and 75/1200).....	£175
U21/22/23 (adds 1200/1200 & tone dial).....	£257
U22 upgrade to existing model.....	£98

All prices include VAT and postage in the UK

12 Bouverie Place, London W2 1RB (tel: 01-724 9053)

Fax: 01-706 2329 Telex: 265251 (ref: 72-0469065) Prestel: 017249053

REBEL ELECTRONICS LTD

AT LAST PERFORMANCE PRODUCTS FOR YOUR QL

RB 100 — WINCHESTER DISC CONTROLLER (Price £195)

- High performance Western Digital Chipset
- Standard ST506/ST412 drive interface
- 8K byte sector buffer
- Plugs into QL expansion port or backplane
- Controls up to two physical drives

RB2000 — CONTROLLER + 20M DRIVE + PSU (Price £399)

- Ready to go system
- Elegant styling

RB50E/RB500E EXPANSION BACKPLANES (£83 or £192 EACH)

ALL PRICES INCLUDE VAT AND P&P

To order: Please fill in coupon and post to

REBEL ELECTRONICS LTD

12 YORK PLACE, LEEDS LS1 2DS. Or telephone

0757 86630/0904 708073

PLEASE HAVE YOUR VISA CARD READY

Please send me:

RB100 ☐ RB2000 ☐ RB50E ☐ RB500E ☐

I enclose cheque — Value £

or debit Visa Card Expiry date:

Signature: Name:

Address: _____

This column introduces and explains the workings of fast, accurate vector graphics commands. PLOT and DRAW work much faster than the standard POINT and LINE, yet they support QL features like multiple windows, 256 stipple colours, over-printing, CTRL-F5 and both screen MODEs.

The PLOT command sets a single dot in any window on the QL screen. It takes two or three integer parameters – an optional channel number, followed by the X and Y co-ordinates of the dot you want to set.

DRAW takes the same parameters – an optional channel, then the X and Y co-ordinates of the end of a line. It draws a straight line joining the last position used with PLOT to the dot at the co-ordinate supplied with DRAW. This command draws a red and green striped line from the top left corner of the screen to the bottom right:

```
WINDOW 512,256,0,0 : INK 4,2 : PLOT
0,0 : DRAW 511,255
```

PLOT and DRAW work in any CONsole or SCReen windows; they use the current INK colour and recognise OVER -1 as well as OVER 0 or 1. OVER -1 combines the line colour and the background in such a way that re-drawing a line in the same colour restores the original background.

PLOT and DRAW use the 'pixel co-ordinate' scheme, explained in the Concepts section of the *QL User Guide*. Co-ordinates are relative to the top left corner of the specified window, as for BLOCK and PIXEL% – DIY Toolkit, June, 1989. This makes the commands much faster than standard QL graphics commands like POINT and LINE.

All QL displays are made up of small dots or pixels, a contraction of PICTURE ELEMENT or PICTURE CELI. The size of QL pixels varies depending on the graphics mode in use.

A MODE 4 screen has 131,072 pixels, made up of 256 lines with 512 pixels on each line. The same co-ordinate scheme is used in MODE 8, allowing X to vary from 0 to 511 and Y between 0 to 255 in a full-screen window but there are only 256 pixels on each line. All X co-ordinates are rounded down to an even value in MODE 8, so lines have the same slope regardless of MODE.

Pixel co-ordinates are integers which relate directly to the address of a point in screen memory, whereas Qdos graphics co-ordinates are floating point numbers which must be multiplied by a scale factor, added to X and Y offsets, and then mapped on to the pixel grid.

If you are drawing a short line, or worse still a single point, these floating point calculations take much more time than the drawing of lines on the screen. It is possible to get some benefit by reducing the precision of floating point calculations to speed the maths but this does not help

DIY TOOLKIT

Simon Goodwin adds fast vector and pixel graphics to the QL repertoire.

much as the procedure is still lengthy. Games and many screen-based drawing programs use integer co-ordinates internally, so they are ideal for use with PLOT and DRAW and needlessly slow with POINT and LINE.

The floating point graphics of the QL are useful when you want to shift or re-scale displays but they are inevitably slow and imprecise. They make it difficult to

align text and graphics together in a window. The shape of pixels varies depending on the TV system in use. Graphics co-ordinates use an extra multiplication factor to correct for this but text is still pixel-based, so displays can overlap unexpectedly on foreign QLs.

Most other computers opt for a simpler and faster co-ordinate scheme, where you supply X and Y dimensions in pixels.

QL World DIY Toolkit September 1989, listing 1.

```
* QL WORLD DIY TOOLKIT - FAST pixel graphics routines
* Version 1.3, Copyright 1989 Simon N Goodwin.
*
xstore    equ    84          Offsets of old X & Y
ystore    equ    78          in the channel block
*
start      lea.l    define,a1
           move.w   $110,a2    BP.INIT vector
           jmp      (a2)
*
draw       lea.l    drawer,a4
           bra.s    get_params
plot       lea.l    plotter,a4
*
* Parameter handler - for 2 or 3 parameters
*
get_params move.w   $112,a2    Vector to get integers
           jsr      (a2)       CA.GTINT
           bne.s    bad_exit
           moveq    #1,d0      First assume channel 1
           subq.w   #2,d3      At least 2 parameters?
           beq.s    get_coords Exactly 2, use #1
           bmi.s    bad_param  Less than 2: an error
           subq.w   #1,d3      Only 1 parameter left?
           bne.s    bad_param  No, too many, complain
           move.w   0(a1,a6.l),d0 Get BASIC channel No.
           addq.l   #2,a1      Discard channel param.
get_coords move.w   0(a1,a6.l),d1 Get X co-ordinate
           move.w   2(a1,a6.l),d2 Get Y co-ordinate
*
* Convert channel number in D0 to ID in A0 and call EXTOP
*
chan_sel    mulu    #40,d0     Channel table size
           add.l    #30(a6.l),d0 Add base offset
```


The QL expects this in cases like WIN-DOW, BLOCK and PIXEL%, so it is useful to have graphics commands which work the same way. In conjunction with their slower ROM brethren, PLOT and DRAW give the QL a set of graphics commands to suit all situations.

PLOT and DRAW check their parameters to ensure that they fit in the specified or default window. The default channel number is #1, so you can re-direct both commands to other channels with USE, the March, 1988 DIY Toolkit offering.

The commands report 'channel not open' if the channel is closed. You get a 'bad parameter' error if you specify the incorrect type of channel or the incorrect number of parameters. 'Error in expression' indicates that one or more parameters could not be evaluated as integers.

PLOT and DRAW return 'not complete' if called when CTRL-F5 has been pressed to pause the display. You do not see this message under normal circumstances, as the operating system traps it, suspends the calling task temporarily and retries until a key is pressed, re-enabling display output.

The new commands should work reli-

ably on any QL or compatible, including the CST Thor and Thor 16. They work only in MODE 8 and MODE 4 but a pixel mask of \$COCO should be sufficient to get the MODE 8 code working in the Thor 16-colour MODE 12, if you can find a way to distinguish that mode from the others.

Ex top

Like earlier DIY Toolkit routines, PLOT and DRAW use SD.EXTOP to add code to the operating system. The workings of EXTOP were explained in the May, 1988 episode of DIY Toolkit, which introduced functions to interrogate the 'channel definition blocks' which store details of channels in use. EXTOP slows things a little but it is a system-independent way to access channel details such as the current window ink, size and position, and the start address of display memory.

The procedure MIRROR, featured in June, 1989, goes substantially faster if you use PLOT rather than BLOCK to set each dot but it is still sluggish by the ROM TRAP handler, which must be navigated twice for each pixel. With PLOT and PIXEL% you now have the code to mirror an entire window with a single EXTOP call, which



would work much faster. Many programmers use BLOCK to draw vertical and horizontal lines, as it is faster than LINE and easier to predict where a block will end. BLOCK is capable of drawing only oblongs, whereas DRAW can cope with diagonals in any direction. Even so, DRAW can be more than twice as fast as BLOCK, without the annoying flicker at the edge of the vertical lines, as BLOCK sets too many pixels and tidies later.

The code for the pixel graphics commands is listed in two forms. Listing two gives you a quick way to enter the code without using an assembler. It loads the equivalent machine code from DATA

cmp.l	#34(a6),d0		bmi.s	neg_DY	
bge.s	what_chan	Past end of table?	addq.l	#1,a3	A3 := SIGN(Delta Y)
move.l	0(a6,d0.l),d0		bra.s	find_DD	
bmi.s	what_chan	Negative if closed	subq.l	#1,a3	
call_extop	move.l	d0,a0	neg.w	d2	D2 := ABS(Delta Y)
move.l	a4,a2	Address of routine	find_DD	cmp.w	d1,d2
moveq	#-1,d3	Allow infinite time	bhi.s	y_bigger	D1 := MAX(D1, D2)
moveq	#9,d0	SD.EXTOP key	no_match	move.w	a2,a4
trap	#3	Call the device driver	sub.l	a5,a5	DDX := DX
rts		Return D0 from EXTOP	bra.s	find_TEMP	DDY := 0
*			y_bigger	exg	d1,d2
bad_param	moveq	#-15,d0	sub.l	a4,a4	D1 := MAX(D1,D2)
rts		BAD PARAMETER error	move.w	a3,a5	DDX := 0
what_chan	moveq	#-6,d0	find_TEMP	move.w	d1,d0
bad_exit	rts	CHANNEL NOT OPEN error	asr.w	#1,d0	DDY := DY
*		Error code is in D0	move.w	d2,d6	TEMP := STEPS DIV 2
* Line DRAW routine			move.w	d1,d2	D6 := PARTS
*			subq.w	#1,d2	D2 is DOT for DBRA
drawer	move.w	*store(a0),d4	bmi.s	null_draw	Exit if line length=0
move.w	ystore(a0),d5	Fetch previous Y	exg	d1,d4	OLDX = D1; STEPS = D4
bsr	range_chk	Line must fit window	exg	d2,d5	OLDY = D2; DOT = D5
find_DX	sub.l	a2,a2			
sub.w	d4,d1	DX := 0	*		
beq.s	set_X_step		* Now draw a line of D5 pixels; D1,D2 = X,Y of next point		
bmi.s	neg_DX		*		
addq.l	#1,a2	A2 := SIGN(Delta X)	dot_loop	add.w	d6,d0
bra.s	set_X_step		cmp.w	d4,d0	TEMP := TEMP + PARTS
neg_DX	subq.l	#1,a2	bcs.s	use_DD	Flag TEMP - STEPS
neg.w	d1	D1 := ABS(Delta X)	sub.w	d4,d0	TEMP := TEMP - STEPS
set_X_step	btst	#3,52(a6)	add.w	a2,d1	OLDX := OLDX + DX
beq.s	find_DY	Check mode (SV.MCSTA)	add.w	a3,d2	OLDY := OLDY + DY
add.l	a2,a2	MODE 4, D1 & A2 are OK	bra.s	do_dot	
ror.w	#1,d1	MODE 8, double X step	*		
find_DY	sub.l	a3,a3	use_DD	add.w	a4,d1
sub.w	d5,d2	And halve X dot count	add.w	a5,d2	OLDX := OLDX + DDY
beq.s	find_DD	DY := 0	do_dot	movem.w	d1-d2,-(a7)
			bsr.s	plot_pixel	Preserve OLDX & OLDY

statements and saves the code in a file. Once you have loaded that file, as follows, you can use PLOT and DRAW in your own programs.

```
base=RESPR(382) : LBYTES "file
name",base : CALL base : NEW
```

The first part of listing two is Marcus Jeffery's standard loader, used in every month's DIY Toolkit project. Only the DATA, from line 590 onwards, changes from month to month.

Listing one is the assembly code program, assembled using HiSoft DevPac. You can type this text into your assembler if you want to customise the code or merge it with other routines. The START routine calls BP.INIT, the ROM vector which adds new commands to SuperBasic. The table labelled DEFINE, at the end of the listing, indicates the names and addresses of the commands. The table is at the end to make it easy to extend the command names if they clash with variables or SuperBasic definitions in your programs.

The rest of the listing can be considered as four parts. The first part is shared between PLOT and DRAW and uses much the same code as PIXEL%, explained in June. It reads the parameters, selects the required channel and calls EXTOP.

The only difference between PLOT and DRAW to this stage is the code vector, copied from A4 to A2, which tells Qdos where to find code to perform the 'EXTENDED OPERATION'. The X and Y coordinates are passed in D1 and D2 respectively.

Supervisor

The remaining three routines are all executed in supervisor mode, as part of the display device driver. The EXTOP call sets two registers to point to important system tables. A6 points to the system variables, whereas it pointed to the start of SuperBasic memory before the call. A0 points to the start address of channel details, as explored in May with my CHAN functions.

The subroutine RANGE_CHK is called by both PLOT and DRAW; it could not be before the EXTOP because it uses the system values of A0 and A6 to check that the operation can be performed.

The first check tests SV.SCRST, the 'SCREEN STATUS' system variable. If this is set all display output is disabled because CTRL-F5 has been pressed. The code returns ERR.NC, the 'not complete' error code. The EXTOP handler sees that the time-out is infinite – in other words it should persevere until the operation can be performed – so it keeps trying, every time a scheduler interrupt occurs, until a key is pressed and SV.SCRST allows output to continue.

Notice the way QUICK_EXIT returns directly to the EXTOP caller, rather than to PLOTTER or DRAWER; it removes the last return address from the stack, so it never returns to the immediately previous caller if an error occurs. The technique saves time and space but it can cause problems unless the hierarchy of calls is defined rigidly.

Next RANGE_CHK ensures that the

movem.w	(a7)+,d1-d2	Restore OLDX & OLDY	plot_pixel	move.l	62(a0),d3	Grab the ink mask
dbra	d5,dot_loop		btst	#0,d2	Find out if Y is odd	
negl_draw	moveq	#0,d0	beq.s	find_word	It's even, mask is OK	
	rte		swab	d3	Odd, so use other mask	
* Do-ord,nate checker; D1=X, D2=Y, ERROR if beyond window						
* Point A1 at the relevant word in video memory						
range_chk	tst.b	51(a6)	Test SV.SCRST	find_word	move.l	50(a0),a1
	bne.s	screen_off	Abort if CTRL-F5 set	lsl.w	#7,d2	1 line= 2^7= 128 bytes
	tst.w	d2	Check Y != 0	add.w	d2,a1	A1 := line RAM start
	bmi.s	range_err		moveq	#7,d2	
	cmp.w	30(a0),d2	Check Y < CH.HEIGHT	and.w	d1,d2	Save X MOD 8 in D2
	bcc.s	range_err		lsl.w	#3,d1	D1 := offset on line
	tst.w	d1	Check X != 0	add.w	d1,d1	Offset := 0-128 (EVEN)
	bmi.s	range_err		add.w	d1,a1	A1 -> screen RAM word
	cmp.w	28(a0),d1	Check X < CH.WIDTH	* Set the pixel in MODE 4 or 8, allowing for OVER -1		
	bcc.s	range_err		* Select pixel mask		
	move.w	#8080,d7	Set MODE 4 pixel mask	set_pixel	move.w	d7,d1
	btst	#3,52(a6)	Check MODE (SV.MCSTA)	ror.w	d2,d1	Align pixel mask
	beq.s	range_ok	MODE 4, no problems	and.w	d1,d3	D3 := 1 pixel of ink
	bclr	#0,d1	MODE 8, X must be EVEN	btst	#3,66(a0)	Check for OVER -1
	move.b	#C0,d7	Mask B colours	bne.s	overplot	
range_ok	add.w	24(a0),d1	Add window offset to X	not.w	d1	Mask out the rest
	add.w	26(a0),d2	Add window offset to Y	and.w	(a1),d1	Fetch the background
	move.w	d1,xstore(a0)	Store end X on SD.XORG	or.w	d3,d1	Insert the pixel
	move.w	d2,ystore(a0)	Store end Y on SD.YORG	move.w	d1,(a1)	Store the combination
	rte			rte		PLOT returns ERR.OK
* Exclusive OR the pixel						
screen_off	moveq	#-1,d0	NOT COMPLETE error	overplot	eor.w	d3,(a1)
	bra.s	quick_exit		rte		
range_err	moveq	#-4,d0	OUT OF RANGE error	* Number of procedures		
quick_exit	addq.l	#4,a7	Return to after TRAP	define	dc.w	2
	rte				dc.w	plot-*
* Pixel PLOT code, uses A0, A6, D7; alters A1, D1, D2, D3						
* End of procs. no fns						
plotter	bsr.s	range_chk			dc.b	4,'PLOT'
	moveq	#0,d0	Preset ERR.OK report		dc.w	draw-*
					dc.b	4,'DRAW'
					dc.w	0,0,0
					end	

QL World DIY Toolkit September 1989, listing 2.

```

100 REMark Sinclair QL World HEX LOADER
110 REMark by Marcus Jeffery & Simon N Goodwin
120 :
150 CLS: RESTORE : READ space: start=RESPR(space)
160 PRINT "Loading Hex..." : HEX_LOAD start
170 INPUT "Save to file...";f$
180 SBYTES f$,start,byte : STOP
190 :
200 DEFine FuNction DECIMAL(x)
210 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
220 END DEFine DECIMAL
230 :
240 DEFine PROCedure HEX_LOAD(start)
290 byte = 0 : checksum = 0
300 REPEAT load_hex_digits
310   READ h$
320   IF h$="*" : EXIT load_hex_digits
330   IF LEN(h$) MOD 2
340     PRINT"Odd number of hex digits in: ";h$
350     STOP
360   END IF
370   FOR b = 1 TO LEN(h$) STEP 2
380     hb = DECIMAL(b) : lb = DECIMAL(b+1)
390     IF hb<0 OR hb>15 OR lb<0 OR lb>15
400       PRINT"Illegal hex digit in: ";h$ : STOP
420     END IF
430     POKE start+byte,16*hb+lb
440     checksum = checksum + 16*hb + lb
450     byte = byte + 1
460   END FOR b
470 END REPEAT load_hex_digits
480 READ check
490 IF check <> checksum
500   PRINT"Checksum incorrect. Recheck data.":STOP
520 END IF
530 PRINT"Checksum correct, data entered at: ";start
560 END DEFine HEX_LOAD
570 :
580 REMark Space requirements for the machine code
590 DATA 382
600 :
610 REMark Machine code data
620 DATA "43FA016434790000","01104ED249FA0054"
630 DATA "600449FA01143479","000001124E926640"
640 DATA "70015543670C6B32","5343662E3031E800"
650 DATA "54893231E8003431","E802C0FC0028D0AE"
660 DATA "0030B0AE00346C16","203608006B102040"
670 DATA "244C76FF70094E43","4E7570F14E7570FA"
680 DATA "4E75382800543A28","004E6100007295CA"
690 DATA "9244670A6B04528A","6004538A4441082E"
700 DATA "000300346704D5CA","E25997CB9445670A"
710 DATA "6B04528B004538B","4442B4416206384A"
720 DATA "9BCD6006C34299CC","3A4B3001E2403C02"
730 DATA "340153426B24C344","C545D046B0446508"
740 DATA "9044D24AD44B6004","D24CD44D48A76000"
750 DATA "615A4C9F000651CD","FFE270004E754A2E"
760 DATA "0033663A4A426B3A","B468001E64344A41"
770 DATA "6B30B268001C642A","3E3CB080082E0003"
780 DATA "0034670808810000","1E3C00C0D2680018"
790 DATA "D468001A31410054","3142004E4E7570FF"
800 DATA "600270FC588F4E75","61B470002628003E"
810 DATA "0802000067024843","22680032EF4AD2C2"
820 DATA "7407C441E649D241","D2C13207E479C641"
830 DATA "082800030042660A","4641C251B2433281"
840 DATA "4E75B7514E750002","FEAA04504C4F5400"
850 DATA "FE9C044452415700","000000000000","*","31773

```

co-ordinates fit inside the window. If not it returns directly to EXTOP with an 'out of range' error. Register D7 is assigned to the 'mask' of a single pixel in MODE 4; this is a binary pattern containing '1's for each bit used to control a single pixel in MODE 4, and '0' for the rest. Later the mask is shifted sideways to select any pixel from a word of display memory.

Another system variable test reads SV.MCSTA, which has bit 3 set if the machine is in MODE 8. If so, the X co-ordinate is made even and an extra bit is added to the mask, as MODE 8 uses three colour bits, against two in MODE 4.

Internally PLOT and DRAW use co-ordinates relative to the top left corner of the screen, rather than the current window, so the code at RANGE_OK adds the window offsets to D1 and D2. Each co-ordinate must be stored somewhere, in case we need to DRAW from it later. If we were re-writing Qdos we could make extra space available in the channel definition but this is not easy once the channel is open. I have cheated by using the least significant bits of SD.XORG and SD.YORG, the floating point co-ordinates of the graphics origin.

These locations are set by SCALE and used to find the screen origin for slow commands like POINT, LINE and CIRCLE. It seems possible to use the locations for those purposes without problems, so long as you do a PLOT to set a sensible 'last position' after each SCALE command.

The least significant bits are used, so that the values stored by PLOT and DRAW make a difference of less than one part in a million to the scale. The offsets have been set up as equates - XSTORE and YSTORE - at the start of the listing, so it is easy to change them if you wish.

The routine to PLOT a pixel is also used by DRAW. First it calls RANGE_CHK, then it sets DO to zero, so PLOT can return with ERR.OK but DRAW can call the rest of the code without corrupting DO. Both routines go through PLOT_PIXEL, which grabs the ink mask from the channel details. This long word contains the screen bit pattern for the current INK colour, the value which would be in screen memory if the whole screen was that colour.

Stipples mean that a particular 'colour' might use different values on alternate screen lines. The low half of the mask contains the bit pattern for even-numbered lines and the high word holds the pattern for odd-numbered lines.

Next PLOT must find the word of video memory which stores the relevant pixel. Each line takes 128 bytes, so the start of memory for the line concerned is found by adding Y star 128 to the start address of video RAM from the channel details. There are 512 possible X co-ordinates on each 64-word line. We find the required word by converting the pixel X co-ordinate, 0-511, into an even offset between 0 and 126.

```

10 BDRAW 0,0 TO 200,100
20 :
100 DEFine PROCedure BDRAW(olDX,olDY,endX,endY)
140 dx=SIGN(endX-olDX) : dy=SIGN(endY-olDY)
160 mx=ABS(endX-olDX) : my=ABS(endY-olDY)
180 IF mx>=my
200   parts=my : steps=mx : ddx=dx : ddy=0
220 ELSE
230   parts=mx : steps=my : ddx=0 : ddy=dy
250 END IF
260 temp=steps DIV 2
270 FOR dot=1 TO steps
280   temp=temp+parts
290   IF temp>=steps
300     temp=temp-steps
310     olDX=olDX+dx : olDY=olDY+dy
320   ELSE
330     olDX=olDX+ddx : olDY=olDY+ddy
340   END IF
350   PLOT olDX,olDY
360 END FOR dot
370 END DEFine _LINE
380 :
390 DEFine FuNction SIGN(v)
400 IF v<0 : RETURN -1 : ELSE RETURN v>0
410 END DEFine SIGN

```

The algorithm is a slightly faster version of that used in PIXEL%. The OVER -1 code just exclusive ORs a combination of the ink and pixel masks into screen memory, whereas the alternative code fetches the background, masks out the relevant pixel, ORs in the new ink and replaces the word.

The routine to draw lines starts by reading the stored co-ordinates for the start of the line; then it calls RANGE_CHK, which returns with the screen-relative co-ordinates of the end of the line in D1 and D2.

The algorithm to generate a straight line is simple in principle but complicated by the need to draw lines in any direction at any angle. The machine code is convoluted and uses all 16 68008 registers, so I wrote the algorithm in SuperBasic for test purposes after I wrote PLOT and before I machine-coded DRAW.

Listing three shows the algorithm in block-structured Basic. The first stage is to find the direction and magnitude of the line, in the X and Y dimensions. DX indicates whether the line moves left (-1), right (+1) or neither left or right (0). DY is the same for vertical movement; in other words, DX and DY indicate the approximate direction of the line, relative to the horizontal and vertical.

MX and MY indicate the number of pixels to be moved vertically or horizontally. The values are always positive, or zero, because of the ABS function. Assuming pixels are square, you can draw a straight line at any angle between 0 and 45 degrees by stepping through a series of points equal to the greater of MX and MY. At each step you advance X or Y,

whichever is greater, by one. Whether you advance the other co-ordinate depends on the slope of the line. Since we want a straight line, the need for an alternate step depends on how long it was since you last took one.

If MX equals MY the angle is 45 degrees and you step one pixel horizontally and vertically at each stage. If either DX or DY is zero, the angle is zero and you just step either horizontally or vertically until you get to the end.

Most lines fall between those two angles. Sometimes you can make the step in DX and DY but sometimes you must just step in the main direction, indicated by the greater of MX and MY.

The program creates two step values for each dimension - DX and DY, plus DDX and DDY for the alternative steps needed to cope with intermediate angles. DDX and DDY hold the 'main' direction. If MX exceeds MY, DDX is the same as DX and DDY is zero. If MY is greater than MX, DDY matches DY and DDX is zero.

The problem is deciding when to step by (DX,DY) and when to use (DDX,DDY). The correct mix is determined by the ratio of MX and MY. If MX is relatively small you must take many vertical steps for every horizontal one.

Since we want a continuous line, the total number of STEPS and points to be plotted is given by the greater of X and Y. The number of jumps out of line, or non-contiguous PARTS, is clearly the lesser of X and Y. For each step the algorithm adds the number of PARTS to a temporary variable. If the result exceeds the total in

STEPS, the algorithm uses DX and DY to find the next point; otherwise it uses DDX and DDY. At the start TEMP is set to half the number of steps, so jumps out of line are spaced evenly.

Any algorithm which works for angles between 0 and 45 degrees can be persuaded to cover a full circle by swapping X and Y and allowing positive and negative steps. This is the role of the SIGN function-calls and the first IF test.

All the variables in listing three could be local integers but I have not written them as such because QLs are likely to crash if you use more than nine locals and parameters in one definition; further integer operations are interpreted more slowly than floating point maths.

Both problems are solved if you compile the procedure but there is little point as it is just an example. If you call BDRAW with variable parameters you should put them in brackets, or the values will be changed on return from the routine. BDRAW works satisfactorily in MODE 4 but plots each point twice in MODE 8.

The machine code version follows the same procedure as the Basic and the comments show Basic variable names for each register. To make the algorithm as fast as possible there is a fairly long set-up procedure, followed by a tight loop to output each pixel once the steps have been calculated. Notice how the X step is doubled for MODE 8 and the pixel mask is pre-set, so there is no need to check the mode as each dot is output.

The code uses fast integer maths throughout and avoids relatively slow multiplication and division instructions by rotating values left and right to scale them by powers of two.

The speed of DRAW depends on the speed of your QL RAM but it should be noticeably faster than LINE, even on the slowest QLs. It would be easy to make the code faster still, by making DOT_LOOP work directly on screen line and column addresses rather than X and Y co-ordinates but that would make the program longer and more difficult to understand.

There are many simple ways you can enhance PLOT and DRAW. A pixel-relative DRAW_R could be added in a few lines and it would be easy to adjust the parameter handler to cope with multiple lines or points. Clipping at the screen boundary could also be added but might be a little more tricky.

Other improvements would involve writing different versions of the main loop for various directions of line, optimising for stipples which do not change between lines and adding support for FILL, which uses undocumented data structures.

Please write to me, care of QL World, if you have found interesting tweaks or short applications for DIY Toolkit code. Send your suggestions if you would like me to explore a specific area in this column, or to implement new and original commands.

One Man's System

■ Bryan Davies looks back on his choice.

Had the QL been more of a commercial success its main competitor might well have been the Amstrad PCW 8256, even though the two machines appear poles apart to some people. After several years of use, both machines are still essentially tools for people who write documents of one kind or another.

Although no programmer — my inclination is always to choose my own bits and pieces — I bought the QL; the choice at the time was QL or BBC. The selection process was rather a hassle but proved worthwhile in the long run.

The display unit did not involve too much head-scratching. Microvitec looked to be the manufacturer to select and it was a question only of what resolution was needed; fortunately, I was convinced early that colour was required and seeing the low-resolution Cub display connected to BBC computers soon made it clear medium resolution was essential for me. The model, bought in October, 1984, was the Cub 653; the designation seems to vary for what looks like the same unit and it is important when buying second-hand to check that the unit in question will display the full QL picture — 85 characters across the SuperBasic screen and the full Quill screen with 80 characters.

Tweaked

Although the screen colour settings have been tweaked twice and there is a slight misregistration of colours in one corner now, I rate the Cub as excellent and have spent nothing on it in almost four years. As one who sits in front of the screen almost every day for many hours it is safe to say looking at it does not cause much discomfort but one has to accept that the eyes usually cause some trouble when you reach the age of 40; a mono screen might be preferable for some people, because the eyes have to focus only at one range.

If the program allows — e.g., *The Editor* — it is worth experimenting with the background and foreground colours to see which combination makes the eyes feel most comfortable. There has been no evident deterioration in the quality of the Cub picture; one factor which should help maintain picture quality is turning down the brightness when the system is not being used and the programs *Q_Switch* and *TaskMaster* both allow this to be done automatically.

The printer causes more pain to people than the other peripherals. Mine has never caused me to have regrets. It is a

Kaga Taxan KP-810, also sold in virtually identical form as the Canon PW1080A. In contrast to many products, this unit is still on sale and costs much the same as it did in 1984. There is an updated version — KP-815 — and a long-carriage one — KP-910. While there are a few printers at less than £200 I doubt whether they would have given such good service. As with the Cub, it has cost nothing beyond the purchase price other than ribbons.

All bouts of strange behaviour have been traced to software problems — or a piece of paper in the belt drive on one occasion. It would certainly be better to have a choice of fonts but I have yet to design any additional ones; the necessary commands and memory are in the printer to enable them. If choosing a printer now I would almost certainly go for a 24-pin type — NEC or Star — at a price not much higher than the Kaga but they could not give better reliability.

It is difficult to estimate how many sheets of paper have gone through a printer through the years but the figure must be in the tens of thousands for mine, with no obvious deterioration in performance. Small, and possibly overlooked, is the serial interface between QL and Kaga. This is the one most users have, the Miracle, and it also has given no trouble.

Disc drives are more in the add-on category. You must have a display and it is difficult to visualise a system without a printer but you can live without discs. That is a tongue-in-cheek remark. There is no way I would have managed to do serious work for almost four years with only Microdrives for company. The dual Mitsubishi 3.5in. drives on my system have been as good as the Cub and the Kaga; not a penny has been spent on repairs to them. The only extra expense has been a disc drive cleaning kit.

The drives are the one-third-height type, which seem to be a much better product than the half-height type, which were around for a time a few years ago and were apparently withdrawn by the manufacturers after giving much trouble. The current "standard" drive is from NEC and it also is generally reliable.

Disc interface and expansion memory problems figure frequently in readers' letters, so it may sound surprising that the PCML 256K external interface and MP 256K internal memory expansion in my original system have been trouble-free. In the never-ending search for more memory, a Trump Card was added to the other system a year ago and it has become almost indispensable but the two QLs on which it has been used have locked up too

many times for comfort — the PCML/MP system is virtually rock-solid in this respect — and no amount of fiddling with components has proved able to stop it.

This is a situation where one has to weigh the merits and demerits and the Trump Card has, to my mind, been the major hardware advance on the QL scene in recent times, so I would not want to use the system without it.

In the ROM port resides the Ice front-end firmware with a mouse attached. This also is in the indispensable category for me but it has not been without its hiccups. The firmware was replaced several times before it became reliable and the mouse has gone sick twice. Occasional weird system behaviour has been attributed to bad contact between ROM and socket, which has required filing of the alignment slot in the ROM PCB connector, smoothing of the solder on some pins and the attachment of the ROM to the QL by a bracket and two screws.

Weird

As for the QL, it cannot be said there have been no failures and no expense; if you use the keyboard a good deal, a new membrane every year or two should be no surprise and a considerable amount of time and money has gone on cartridges and, to a lesser extent, Microdrives. If you are reasonably adept at doing repairs it will be time rather than money which is spent. Most of my time went into stopping lock-ups and overheating; lock-ups still occur on one system but not on the other and over-heating is a thing of the past since modifying the 5V regulator set-up, drilling ventilation holes and using one of Dennis Briggs' power supplies.

It is very easy to give advice on how to keep a system working trouble-free but it is much more difficult to offer convincing arguments why particular precautions enhance reliability. Because it seems common sense to do so. I frequently vacuum the computer system area and room, using a Black & Decker car vacuum cleaner, and wipe the table and system units with a damp cloth. Dust and dirt are brushed from the computer, disc drives and printer with a soft paste brush, also frequently and prior to vacuuming. The drive cleaner is used every three weeks. Disc errors are virtually non-existent.

The screen-dimming function ensures that the phosphor coating on the inside of the tube gets minimum excitation for most of the time the system is on; although the system is on, typically, 10-15 hours per day, it is rare for me to be putting anything on the screen for more than half that time.

So what of the future? The only new piece of hardware which looks to be on the horizon is a hard disc drive. A hard disc can transform a system for a user who makes heavy demands on it. You soon get used to moving megabytes in seconds.



SUPER BASIC

Unlike Unix and MS-DOS, Qdos does not include a pattern-matching facility. Mike Lloyd rectifies that anomaly with a powerful SuperBasic function.

Many SuperBasic projects start as a result of thinking "Would it not be pleasant if". Would it not be pleasant if the DIR command could be asked to list only those filenames suffixed with "_doc". It would also be good if a single DELETE command could remove all files associated with "test", such as "test1_bas", "oldtest1_bkp" and "newtest_doc".

For once, the MS-DOS operating system is ahead of Qdos because its file management commands allow users to specify groups of files related by filename. The Unix operating system is even more advanced, because its pattern-matching algorithms work not only with file names but also with file contents. Unix equivalents to the suggestions above are:

ls ★_doc (ls = list)

rm ★_test★ (rm = remove)

Pattern matching is also important in database applications because it allows records which share common characteristics to be grouped. An inventory database might be searched for all types of bolts, or a mailing list searched for all London addresses, or a data file examined to retrieve lines which include the initials QL.

Wildcards

Many QL owners already have a simple file name matching facility provided either by Tony Tebby's Toolkit or by their floppy disc interfaces but it cannot distinguish between the suffixes "_doc" and "_docy" and it would not associate "newtest_doc" with other "test" files. Nor can it be used to search file contents.

The most important element of pattern matching, and what Tebby's Toolkit lacks,

is *wildcards*, or characters with special meanings. The term is derived from card games in which a particular card can be used to represent any other card. Two wildcards are implemented in the SuperBasic function listed with this article, the asterisk and the question-mark.

When an asterisk appears in a pattern string it represents any number of characters, including none at all. The pattern "Fred★doc" therefore will match "Fred_bkp_doc", "Freda.doc" and "Freddoc". The question-mark represents any single character. Thus, "Fred?"

"Many SuperBasic projects start as a result of thinking 'Would it not be pleasant if . . . ?'"

matches any five-letter string which begins with the letters "Fred". "Freda" and "Fredi" would match, while "Freddy" and "Fred" would not.

Having used the asterisk and question-mark as wildcards it seems that they cannot then be used with just their ordinary meanings. This problem is solved by a third metacharacter, the backslash (\), which can be used before either of the other characters to *dereference* it, or take away its special meaning. For instance, all strings which end in question-marks can be selected using the pattern "★I?".

Wildcards offer powerful benefits which more than offset the sometimes awkward construction of patterns. The following examples presume the existence of a dictionary array:

"T???" finds all four-letter words beginning with "T"

"★ing" finds all words ending in "ing"

"★j★" finds all words containing a "j"

"???★" finds all words longer than three letters

"Dar?ing" finds Darling, Darning and Darting

"Dar★ing" also finds Daring, Darjeeling, Darkening and so on.

A pattern string comprises groups of ASCII characters interspersed by groups of wildcards. To match a pattern with a target string each group of ASCII characters must find exact equivalents at appropriate points in the target, determined largely by whatever wildcards appear immediately before them in the pattern string. A simple algorithm which suggests itself is therefore:

"Use the INSTR function to match each text group in turn with the part of the target string dictated by previous matches and wildcards until either a complete match is found or until the end of the target string is reached."

This algorithm does not always work. Imagine a search using the pattern "★S?N★". The algorithm will identify "Sinclair" correctly as a match but it would reject the equally correct "Sir Clive Sinclair" because the INSTR function would find the first "S" in "Sir", discover that the second letter following is not an "n" and report a failure, never realising that a correct match can be found later in the string.

If a single pass of the target string is unreliable, some form of recursion is needed. Recursive algorithms are those defined in terms of themselves and

Listing 1

```

100 DEFine FuNction Match (Pattern$, Str$)
102 LOCAL Char, Text, Wcard, Mask$, Loop, Result
104 LOCAL Charpos(10), Endchar(10), First(10)
106 LOCAL MinPos(10), MaxPos(10), X, Y
108 Mask$ = Pattern$: Char = 1: Text = 0
110 FOR X = 1 TO 10
112   REPEAT Loop
114     Wcard = Mask$(Char) INSTR "\?*"
116     IF Wcard>1 AND Text: Text = 0: EXIT Loop
118     IF ((Wcard>1) + Text) = 0
120       Text = 1: Charpos(X) = Char
122     END IF
124     SElect ON Wcard
126     = 1: IF Char = LEN(Mask$): STOP
128       IF Char = 1
130         Mask$ = Mask$(2 TO)
132       ELSE
134         Mask$ = Mask$(1 TO Char-1) &
          Mask$(Char+1 TO)
136       END IF
138     = 2: MinPos(X) = MinPos(X) + 1
140     = 3: MaxPos(X) = 1
142   END SElect
144   Char = Char + 1
146   IF Char > LEN(Mask$): EXIT Loop
148 END REPEAT Loop
150 Endchar(X) = Char - 1
152 IF Char > LEN(Mask$): EXIT X
154 END FOR X

```

Listing 2

```

200 Y = 1: First(Y) = 1 + MinPos(1)
202 REPEAT Loop
204   IF Charpos(Y) = 0
206     IF MaxPos(Y)
208       Result = LEN(Str$) >= First(Y) - 1
210     ELSE
212       Result = LEN(Str$) = First(Y) - 1
214     END IF
216   ELSE
218     IF First(Y) > LEN(Str$): RETURN 0
220     IF MaxPos(Y)
222       Last = LEN(Str$)
224     ELSE
226       Last = First(Y) + Endchar(Y) - Charpos(Y)
228     END IF
230     Result = Mask$(Charpos(Y) TO Endchar(Y)
          INSTR Str$(First(Y) TO Last)
232   END IF

```

Listing 3

```

300 IF Result = 0
302   REPEAT backtrack
304     Y = Y - 1: IF Y = 0: RETURN 0
306     IF MaxPos(Y): EXIT backtrack
308   END REPEAT backtrack
310 ELSE
312   IF Charpos(Y) = 0: RETURN 1
314   First(Y) = First(Y) + Result
316   First(Y + 1) = First(Y) + Endchar(Y)
          - Charpos(Y) + MinPos(Y + 1)
318   Y = Y + 1
320 END IF
322 END REPEAT Loop
324 END DEFine

```

understanding them can confuse even the most experienced programmers. To illustrate recursion, Confucious might have said:

A journey is a journey followed by a single step.

The great philosopher would have been on to something except that the definition is invalid for the first and last steps in the journey. Recursive computer functions are usually started by an externally-provided parameter – the first step – and completed when some condition is reached – the last step. About the simplest example is a function to compute

"The great philosopher would have been on to something, except that the definition is invalid for the first and last steps."

factorials. The factorial of 5 is $5 \times 4 \times 3 \times 2 \times 1$, or 120. A SuperBasic function to compute the factorial of any number is:

```

100 DEFine FuNction Factorial (num)
110 IF num = 1 THEN
120 RETURN 1
130 ELSE
140 RETURN num * Factorial (num - 1)
150 END IF
160 END DEFine Factorial

```

A recursive approach to pattern matching requires a new definition of a pattern string which divides it into a series of like elements. The standard element of a pattern string comprises at least one wildcard followed by a number of "normal" characters. Exceptions to the standard are elements comprising only wildcards or text. Patterns can contain any number of elements "flp?_*doc" consists of the elements "flp", "?_" and "*doc".

A recursive algorithm would take each element separately and attempt to find a match. If successful, matches for succeeding elements would be attempted until the last element in the pattern was reached. If any match failed the algorithm would back-track to find a further match for the *previous* element, if there was one.

This is best illustrated by an example. The target string "The rain in Spain" might be checked against the pattern "*ain". The text characters, "ain", are sliced from the pattern string and a match is found in "rain". Without a trailing asterisk the match must occur at the end of a string,

which in this case is not true. The function then repeats itself with what remains of the target string, "in Spain" and this time finds a successful match in "Spain".

If the target string is "The rain in Portugal" the first match with "rain" would again fail and a second run through the algorithm would fail to find another occurrence of "ain" and so the algorithm would report a failure.

Unfortunately, although SuperBasic allows recursive functions to be defined, the one written to accompany this article persistently corrupted the Qdos name table which stores details of all SuperBasic keywords and user-defined variables. The cause and cure for this problem remain unidentified. Strangely, the function runs properly when compiled with Turbo and so some rare SuperBasic bug is possibly to blame.

To complete the assignment on time the function was re-written completely using arrays of pointers in the place of recursion and it is this somewhat monolithic routine which is spread across listings one, two and three.

Listing one begins by declaring all the necessary local variables and arrays. Three of the arrays are used for pointers. Charpos and Endchar are used to hold the positions of the first and last characters in each group of normal characters. The First array shows the first position in the target string from which each search may start.

Variables

The Minpos array records how many question-mark wildcards appear in each pattern element. The Maxpos array shows whether or not an asterisk wildcard is present. Other variables will be explained as they are used.

A local copy of the pattern string, *Mask\$*, is obtained so that it can be sliced without affecting the parameter. A pointer to the "current" character is set to one and a flag indicating whether text or wild-cards are being processed is set to zero, indicating that a wildcard is next expected. A FOR...NEXT loop repeats for each pattern element. Recalling the definition of a pattern element it can be seen that the pattern "★??Sinclair" comprises a single element, whereas "?S★r" contains two. It is unlikely that more than 10 elements will occur.

In the FOR...NEXT loop a REPEAT loop tests each character in *Mask\$* against a string of metacharacters. The two IF statements which follow at lines 116 and 118 test for what astrologists might call the cusp between groups of adjacent wildcards and groups of text characters. If a wildcard following a run of normal characters is detected, it belongs to the next pattern element and so the loop is exited. On the other hand, if the first text character in an element is detected its position is recorded in the

Listing 4

```
400 REMark *** Example Use of Match
402 REPEAT SongMatch
404   INPUT#0; "Match = "; Test$
406   IF Test$ = "": EXIT SongMatch
408   CLS: PAPER 0: INK 7: RESTORE 400
410   FOR Try = 1 TO 10
412     READ Song$: k = Match (Test$, Song$)
414     IF k: a$ = "HIT": ELSE : a$ = "MISS"
416     STRIP 2*k: PRINT a$; TO 6; Song$
418   END FOR Try
420 END REPEAT SongMatch
422 DATA "Here, There and Everywhere"
424 DATA "And I Love Her", "True"
426 DATA "Dancing in the Dark", "Private Dancer"
428 DATA "Loving You", "It Must Be True"
430 DATA "There Always Will Be You"
432 DATA "Dance Away", "Will You?"
```

Charpos array and the text flag is set to one before the REPEAT loop continues.

Wildcards alter the bounds of the search area in the target string. A pattern of "★doc★" will search the entire target string for the occurrence of the characters "doc". It does not matter how many consecutive asterisks appear; the effect is still the same. With question-marks, however, quantity is important and so they are counted in the SELECT structure.

The effect of the backslash, which dereferences a following wildcard, is produced by slicing the pattern string to

search this point moves rightwards through the target string. The initial First value is set according to how many question-mark wildcards appear in the first pattern element.

A feature of the pattern analysis is that the Charpos value for the last pattern element will always be zero. Using this to identify the last element. Result can be set according to how many characters remain in the string.

For elements other than the last, an end point for the search must be calculated. If an asterisk is present it will be the final character in the target string; otherwise it depends on the length of the current group of text characters and the presence of question-mark wildcards.

Bounded

Unless the First character position is beyond the end of the target string, the part of the target string bounded by the First and Last character positions is searched for a match at Line 230.

Listing three acts according to the value of Result. If Result is zero the backtracking loop returns to the last pattern in which an asterisk appears to try for a further search.

If Result is positive the current First value is incremented by the Result value. The First value for the next pattern element is then calculated based on the location of the match, the length of the current normal character group and the existence of question-mark characters in the next pattern element. Finally, the Y counter is incremented before the loop repeats.

If the code remains confusing, do not worry. Type-in the function and test it using the fourth listing against either the data provided or against your data.

● Next month Mike Lloyd puts pattern matching to serious use in a file management utility.

"The Unix operating system pattern-matching algorithms work not only with files names but also with file contents."

remove it. This ensures that the wildcard shifts to the left by one place and thus avoids detection at the beginning of the next loop.

The REPEAT loop ends by incrementing the Char pointer. It is exited at the end of each pattern element, at which point the Endchar pointer is saved and the FOR...NEXT loop is cycled. When the final element has been processed control passes to the next listing.

The object of the second listing is to identify whether a match exists between the text characters in a pattern element and the area of the target string currently under scrutiny. A new counter, Y, controls progress through the pattern elements.

The values of the First array vary to show the point in the target string at which each search must begin. After each

P+R:O=G<S

If you have a program worthy of consideration, send it to 'The Progs',
Sinclair QL World, Greencoat House, Francis Street, London SW1P 1DG.
We pay for everything published at the usual page rates.

Program of the month

WORDBLOK by Phillip Sproston

Wordblok is simple to play but difficult to win. All you have to do is choose a word grid from one to 10, or let the computer select a random grid for you. You will then be asked to find

as many words as you can make from the word grid; the catch is that every word must include the letter on the right of the grid.

All words are of four letters or more and each grid contains at least one word with 10

letters. No foreign words, proper names or plurals are allowed; remember that a plural does not always have an 's' on the end.

At the end of each round the computer will fill in the missing words and tell you your score.

You can cheat and look at the listing for the answers but if you really want a testing time, get somebody else to type in the data for you. Then you could develop some more grids and give your friends a hard time.

mdv1_wordblok 1961 Jul 14 23:49:42 Fri

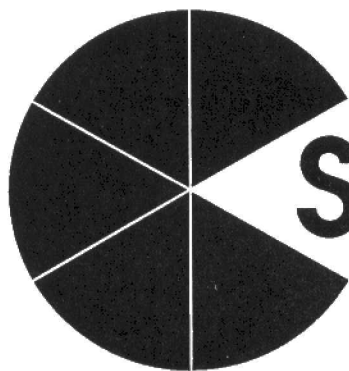
```
100 REMark WORDBLOK
110 WINDOW 448,200,32,16:PAPER 0:WINDOW £2,448,200
,32,16:PAPER £1,0:MODE 2:OPEN £8,CON_:WINDOW £8,51
2,255,0,0:PAPER £8,0:OG=0
120 CLS £8:CSIZE 2,1:INK 7
130 AT 1,13:PRINT 'WORDBLOK'
140 A$= QUANTUM SOFT 1985:INK 4
150 FOR C=1 TO LEN(A$)-1
160 AT 4,8:PRINT A$(LEN(A$)-C TO ):BEEP 0,C,C,1,2,
3,4,6
170 END FOR C:BEEP:INK 2
180 AT 7,4:PRINT "PRESS 'I' FOR INSTRUCTIONS"
190 AT 9,8:PRINT "PRESS 'S' TO START"
200 IF KEYROW(3)=8 THEN GO TO 500
210 IF KEYROW(5)=4 THEN GO TO 260
220 GO TO 200
260 CLS £8
270 CSIZE 3,1:AT 0,9:INK 2:UNDER 1:PRINT 'WORDBLOK
':UNDER 0:INK 7
280 CSIZE 1,0:PRINT "\"THE OBJECT OF 'WORDBLOK' IS
TO MAKE AS MANY WORDS AS POSSIBLE OUT OF THE LE
TTERS PROVIDED."
290 PRINT "\"THE WORDS MUST BE AT LEAST FOUR LETTER
S LONG AND EACH OF THE LETTERS CAN ONLY BE USED ONC
E."
300 PRINT "\"HOWEVER EACH WORD MUST CONTAIN THE LAR
GER LETTER ON THE RIGHT OF THE OTHER LETTERS."
310 PRINT "\"THERE IS AT LEAST ONE 10 LETTER WORD."
320 PRINT "\"NO PLURALS ARE ALLOWED, NO FOREIGN WOR
DS, AND NO PROPER NAMES(e.g ANDREW).":INK 2:PRINT
"
TYPE '*' FOR YOUR FINAL SCORE.":INK 7
```

```
330 CSIZE 3,1:INK 4:AT 9,2:PRINT "PRESS 'ENTER' TO
START"
350 K$=INKEY$(5):IF CODE(K$)<>10 THEN GO TO 350
500 CLS£8:INK 4
510 CSIZE 2,1:AT 2,4:UNDER 1:PRINT 'PLEASE PRESS K
EY FOR OPTION':UNDER 0
520 INK 2:AT 5,5:PRINT "'C'=CHOOSE A GRID NUMBER"
530 AT 7,5:PRINT "'R'=A RANDOM GRID NUMBER"
540 IF KEYROW(2)=8 THEN GO TO 580:REMark CHOOSE
550 IF KEYROW(5)=16 THEN GO TO 650:REMark RANDOM
560 GO TO 540
580 CSIZE £0,3,1
590 CLS£0:PRINT£0,'PRESS GRID NUMBER (1 TO 10)
600 G$=INKEY$(5):IF G$=' ' THEN GO TO 600
610 IF G$>'9' OR G$<'0' THEN GO TO 600
620 IF G$='0' THEN G=10:GO TO 1000
630 G=G$
640 GO TO 1000
650 K$=INKEY$(9):IF K$<>' ' THEN GO TO 650
660 RANDOMISE:G=RND(1 TO 10):IF G=OG THEN GO TO 66
0
670 OG=G
999 REMark SET UP
1000 CLS£8:CSIZE 3,1:INK £0,2:CSIZE£0,3,1:UNDER 1:
INK 4:AT 1,18:PRINT 'WORDBLOK':UNDER 0:INK 7:RESTO
RE (10000+(10*(G-1)))
1010 READ N,W$,A,B,E
1020 DIM T$(N,10):DIM P$(N,10):FOR C=1 TO N
1030 READ T$(C):END FOR C
1050 PAPER 2:FOR C=0 TO 2
1060 AT 3+C,20:PRINT W$((3*C)+1 TO (3*C)+3):END FO
R C:PAPER 4
1070 AT 3,23:PRINT ' ':AT 4,23:PRINT W$(10):AT 5,2
3:PRINT ' ':PAPER 0
```

```

1080 C$=C:IF C>20 THEN AT C-21,20-(LEN(C$)):GO TO
1120
1100 AT C-1,4-LEN(C$)
1120 PRINT C$;','
1150 END FOR C
1160 AT 13,36:INK 2:UNDER 1:PRINT 'OTHERS':UNDER 0
1170 AT 4,35:PRINT QUANTUMSOFT 1985":AT 0,41:PR
INT 'GRID:':G:AT 7,33:PRINT 'WORDS':INK 7:CSIZE 3,
1:AT 4,17:PRINT 'O':CSIZE 1,0
1180 K$='* FOR FINAL SCORE':FOR C=1 TO LEN(K$)
1190 AT C+1,54:PRINT K$(C):END FOR C
1200 DIM O$(6,10):REMARK OTHERS
1210 S=0:REMARK NUM OF WORDS
1220 XS=0:REMARK XTRA WORDS
1500 CLS$:INPUT £0,"      WORD:":E$:IF E$='' THEN
GO TO 1500
1505 IF E$(1)='*' THEN GO TO 3000
1510 L=LEN(E$):IF L<4 OR L>10 THEN
1520 CLS$:PRINT £0,"      WRONG WORD LENGTH"
1530 BEEP 2000,99:PAUSE 50:GO TO 1500
1540 END IF      FOR C=1 TO L
1545 IF E$(C)=' ' THEN CLS$:PRINT £0:"      SPACES
NOT ALLOWED":BEEP 2000,50:PAUSE 50:GO TO 1500
1550 END FOR C:C$=W$
1560 FOR C=1 TO L
1570 IF E$(C)=W$(10) THEN GO TO 1600
1580 END FOR C:CLS$:PRINT £0,"      NO LETTER '";
W$(10);""
1590 BEEP 2000,80:PAUSE 50:GO TO 1500
1600 FOR C=1 TO L
1610 FOR V=1 TO 10
1620 IF E$(C)=C$(V) THEN GO TO 1640
1630 END FOR V:CLS$:PRINT £0,"      LETTER ERROR IN
WORD":BEEP 2000,70:PAUSE 50:GO TO 1500
1640 C$(V)=' ':END FOR C
1670 FOR C=1 TO N
1680 IF T$(C)=E$ THEN GO TO 1800
1690 END FOR C:FOR C=1 TO 6
1695 IF E$=O$(C) THEN AT 13+C,36:INK 4:PRINT E$:BE
EP 2000,40:CLS$:PRINT £0,"YOU ALREADY HAVE THAT O
NE!":PAUSE 80:INK 7:AT 13+C,36:PRINT E$:GO TO 1500
1700 END FOR C:BEEP 2000,40:CLS$:PRINT £0," I DO
N'T KNOW THAT WORD"" PLEASE CONFIRM (Y OR N)"
1710 K$=INKEY$(5):IF K$='' THEN GO TO 1710
1720 IF K$='N' THEN GO TO 1500
1730 IF K$<>'Y' THEN GO TO 1710
1740 CLS$:PRINT £0,"      OK I BELIEVE YOU!":BEEP
2000,40:PAUSE 45
1750 XS=XS+1:IF XS>6 THEN XS=6:AT 19,36:PRINT '
'
1760 O$(XS)=E$:AT 13+XS,36:PRINT E$:S=S+1:GO TO 20
00
1800 FOR C=1 TO N
1810 IF P$(C)=E$ THEN
1820 BEEP 2000,30:CLS$:PRINT £0," YOU HAVE ALREAD
Y GOT THAT!"
1830 C$=C:IF C>20 THEN AT C-21,21:GO TO 1850
1840 AT C-1,5
1850 INK 4:PRINT E$:PAUSE 50:INK 7
1860 C$=C:IF C>20 THEN AT C-21,21:GO TO 1880
1870 AT C-1,5
1880 PRINT E$:GO TO 1500
1890 END IF
1900 END FOR C
1902 FOR C=1 TO N
1905 IF T$(C)=E$ THEN GO TO 1910
1907 END FOR C
1910 CLS$:PRINT £0,"      THAT'S CORRECT":BEEP 20
00,60
1930 IF C>20 THEN AT C-21,21:GO TO 1950
1940 AT C-1,5
1950 PRINT E$
1960 P$(C)=E$:S=S+1:PAUSE 40
2000 AT 0,0:S$=S:CSIZE 3,1:AT 4,18-LEN(S$):PRINT S
:AT 0,0:CSIZE 1,0:GO TO 1500
3000 CLS$:PRINT £0,"      GAME ABANDONED"
3010 INK 2:FOR C=1 TO N
3020 IF T$(C)<>P$(C) THEN
3030 IF C>20 THEN AT C-21,21:GO TO 3050
3040 AT C-1,5
3050 PRINT T$(C)
3060 END IF
3070 END FOR C:INK 7
3080 CLS$:PRINT £0,"      PRESS 'ENTER' WHEN READY"
3090 K$=INKEY$(9):IF K$='' THEN GO TO 3090
3100 IF CODE(K$)<>10 THEN GO TO 3090
3110 CLS£8
3120 K$='FINAL RATING':CSIZE 3,1:INK 4:FOR C=1 TO
LEN(K$)-1
3130 AT 3,7:PRINT K$(LEN(K$)-C TO ):END FOR C
3140 INK 2:R$='POOR'
3150 IF S>=A THEN R$='GOOD'
3160 IF S>=B THEN R$='VERY GOOD'
3170 IF S>=E THEN R$='EXCELLENT'
3180 AT 9,1:PRINT "PRESS 'SPACE' TO CONTINUE"
3190 FOR C=0 TO 7 STEP 2
3200 INK C:AT 5,13-(LEN(R$)/2):PRINT R$
3210 K$=INKEY$(9):IF K$=' ' THEN INK 7:CLS £8:GO T
O 120
3220 END FOR C:GO TO 3190
8888 CSIZE £0,0,0
9998 STOP
9999 DELETE MDV1_WORDBLOK:SAVE MDV1_WORDBLOK:STOP
10000 DATA 32,'THEERONERD',23,27,32,'CEDE','CENTRE
D','CHEERED','CREED','DECENT','DECREE','DEER','DEN
E','DENT','DETER','DRENCH','ENTERED','ERECTED','ER
RED','ETCHED','HEED','HERD','NEED','RECEDE','REED'
,'REND','RENDER','RENTED','RETCHED','RETRENCHED','
TEED'
,'TEHEED','TEND','TENDER','TREED','TRENCHED','TREN
D':REMARK 14 MAR
10010 DATA 26,'BADLYNTANU',16,21,26,'ABUNDANT','AB
UNDANTLY','ABUT','ADULT','ANNUAL','ANNUL','AUNT','
BAUD','BLUNT','BUNA','BUNT','BUTYL','DAUB','DAUNT'
,'DUAL','DULY','DUTY','LANDAU','LAUD','TABU','TUBA
','TUBAL','TUNNY','UDAL','ULNA','UNLAY'
10020 DATA 26,'NICOTNOROT',20,23,26,'CITRON','CONT
ORT','CONTORTION','COOT','COTTON','CROTON','INTO'
,'NOTION','ONTO','OTIC','OTTO','RIOT','ROOT','TINCT
','TINT','TIRO','TONIC','TOOT','TORC','TORIC','TOR
N','TORT','TRICOT','TRIO','TRITON','TROT'
10030 DATA 33,'ACEYONVECN',22,27,33,'ACNE','AEON'
,'ANCON','ANNOY','ANON','ANYONE','CANE','CANNY','CA
NOE','CANON','CANYON','CONCAVE','CONE','CONEY','CO
NVENE','CONVEY','CONVEYANCE','COVEN','CYAN','ENVOY
','ENVY','EVEN','NAVE','NAVY','NEON','NONCE','NONE
','NO
VA','NOVENA','OCEAN','ONCE','OVEN','VANE'
10040 DATA 40,'TONICEMMMT',29,35,40,'CENT','CENTO'
,'CITE','COMET','COMMENT','COMMIT','COMMITMENT','C
OTE','EMIT','ENTOMIC','INTO','ITEM','MINT','MITE'
,'MITT','MITTEN','MOMENT','MOTE','MOTET','MOTTE','N
ETT','NOETIC','NOTE','NOTICE','OCTET','OMIT','OTIC
','TE
NT','TICE','TIME','TINET','TINE','TINT','TOME','TO
NE','TONEMIC','TONIC','TOTE','TOTEM','TOTEMIC'
10050 DATA 25,'ROBIYNIHIT',17,21,25,'BIOTIN','BIRT
H','BORT','BOTH','BROTH','HINT','INHIBIT','INHIBIT
OR','INHIBITORY','INTO','NORTH','OBIT','ORBIT','RI
OT','THIN','THORN','THORNY','THROB','TINY','TIRO'
,'TOBY','TORN','TRIO','TROY','TYRO'
10060 DATA 39,'OILRIGNIER',28,33,39,'ENROL','ERRIN
G','GIRL','GIRO','GOER','GORE','GRIN','GROIN','IGN
ORE','INLIER','IRON','IRONER','IRRELIGION','LIGER'
,'LIGROIN','LINER','LINGER','LOIR','LONER','LORE'
,'LORINER','LORN','NEROLI','OGRE','OILER','ORIEL','
ORIGI
N','ORLE','REGION','REIGN','REIN','RELIGION','RILE
','RILING','RING','RINGER','ROIL','ROILING','ROLE'
10070 DATA 40,'EATNIBLIEV',30,35,40,'ALIVE','ANVIL
','BEVEL','ENVIABLE','EVEN','EVENT','EVIL','INEVIT
ABLE','INVITE','INVITEE','LAVE','LEAVVEN','LENITIV
E','LEVANT','LEVIN','LIVE','LIVEN','NAIVE','NATIVE'
,'NAVE','NAVEL','VAIL','VAIN','VALE','VALET','VAN
E','V
EAL','VEIL','VEIN','VEINLET','VELETA','VENAL','VEN
IAL','VENT','VENTIL','VIALE','VIAL','VILE','VINE'
,'VITAL'
10080 DATA 28,'REQUESTING',19,24,28,'ENQUIRE','EQU
INE','ESQUIRE','INQUEST','QUEEN','QUEER','QUEERING
','QUERIST','QUERN','QUEST','QUESTING','QUIET','QU
IETEN','QUINT','QUINTE','QUIRE','QUIT','QUITE','QU
ITS','REQUEST','REQUESTING','REQUIRE','SEQUENT','S
EQUIN
','SQUINT','SQUIRE','SQUIREEN','SQUIRT'
10090 DATA 40,'LAPYIBCEMC',29,35,40,'ACME','AMICE'
,'BECALM','BICE','BICYCLE','CABLE','CALM','CAME','
CAMEL','CAMP','CAPE','CELIBACY','CICELY','CLAIM','
CLAM','CLAMP','CLAP','CLAY','CLIMB','CLINE','CLIP'
,'CYCLE','CYMBAL','EPIC','EPICAL','IMPECCABLY','LA
CE','
LACY','LAIC','MACE','MACLE','MALIC','MALICE','MELI
C','MICA','PACE','PICA','PLACE','PLAICE','PYAEMIC'

```

Sector Software

The best programs and peripherals for the QL

Overdrive

Printer driver for any program. 255 translate sequences, will print screendumps within standard files e.g. Quill documents.
Expanded QL Only. **£16**

QZ/QL to Z88 file transfer

Software and cable to connect the Z88 and QL and transfer any files between them. Includes Archive to Pipedream and back conversion routines. **£25**

Spellbound

A spelling checker that checks your spelling AS YOU TYPE. Based on a 30,000 word dictionary, works with Quill or The Editor V1.17 onwards on the expanded QL. **£30**

Image processor

Tidies up images produced by a digitiser. Will produce clip art for PD2 from QL screens and picture enhancement. **£19**

Taskmaster

A brilliant multitasking front end system which lets you use the QL as a serious machine. Multitask many programs at once. **£25**

Sinclair Satellite

We now have Clive Sinclair's satellite TV systems in stock. Only **£195** for a full 48 channel, stereo, remote control version. Phone now for details.

Write Turn

Turn spreadsheets and documents on their sides with this excellent utility, works on Epson and compatible printers **£12**

QL World Index

A complete index to the contents of QL World from its start to May 1988. Find articles and reviews in seconds, 160K+ of data compressed to fit into a 128K QL **£6**

Flashback

A very fast and slick database which has very few limitations. Will also convert Archive files. **£25**

Flashback — Special Edition

Many new features over the original flashback inc report generator etc, etc. **£40**

Touch Typist

Excellent typing tutor that works. 200 lessons, graph of your progress, adjustable difficulty levels **£12**

New Enhanced QL Test software & lead

Will test your major QL functions. Includes an RS232 loopback lead to test the serial ports. **£14**

Ferret

Find lost files fast with this file search utility which will read all your files on disk or mdv looking for a match with your search text. **£12**

STD Index

This index to all the dialling codes in the country executes from disk in 15 seconds. Know the place and it will tell you the number, know the number and it will tell you the place! (Expanded QL only.) **£12**

Page Designer 2

This is a full feature desktop publisher that has to be seen to be believed. Ask for full details of this system and its support programs. **£35**

Phillips CM8833 Colour Stereo Monitor

A stereo monitor for the QL, Amiga, ST or almost any computer **£260**

MDV Cartridge Doctor

Rescues corrupt files and microdrives	£13
Tandata Qcom Stacking Modem	£39
LC10 printer	£229
LC10 colour printer	£274
LC2410 printer	£374
QL Service Manual	£25
Keyboard membranes only	£6
ZX8301 ULA	£12
Microdrive cartridges	£1.90
3.5" DSDD disks	£1.25

Massive reductions in Z88 prices!

Z88 Computer	Was £287
	Now £230

Z88 All in one kit:

Z88 computer, 128K RAM, Mains Adaptor, custom carry case, Batteries, User Manual	Was £329
	Now £287

If you are not receiving our free QL catalogue just send your name and address and we will include you in future mailings

QL SuperBasic

The definitive handbook by Jan Jones

£8



Sector Software



39 Wray Crescent, Ulmes Walton, Leyland, Lancs PR5 3NH
Tel 0772 454328, Fax 0772 454680, Prestel Mbx 772454328
All prices include VAT and P&P